Universidad Carlos III de Madrid

Escuela Politécnica Superior

Ingeniería Técnica de Telecomunicación:
Sistemas de Telecomunicación

Proyecto Fin de Carrera

Implementación de QoS en la parte de acceso WLAN en una red 4G

Autor: Miguel Ángel Rodelas Delgado
Tutor: Dr. Pablo Serrano
Cotutor: Dr. Albert Banchs

Octubre 2007

PROYECTO FIN DE CARRERA

Departamento de Ingeniería Telemática

Universidad Carlos III de Madrid

Título: Implementación de QoS en la parte de acceso WLAN en una red 4G

Autor: Miguel Ángel Rodelas Delgado

Tutor: Dr. Pablo Serrano

Cotutor: Dr. Albert Banchs

La lectura y defensa del presente proyecto fin de carrera se realizó el día 2 de Octubre bajo el tribunal:

- **Presidente:** Dr. Carlos Jesús Bernardos
- **Secretario:** Dr. Mario Muñoz Organero
- **Vocal:** Almudena Lindoso Muñoz

Habiendo obtenido la calificación de:

Presidente **Secretario** **Vocal**

"Vive tu vida como si subieras una montaña. De vez en cuando mira hacia tu alrededor y admira las cosas bellas en el camino. Sube despacio, firme y disfruta cada momento hasta llegar a la cumbre."

Harold V. Melchert

Agradecimientos

Este proyecto es el resultado de un largo camino que no hubiese sido capaz de recorrer sin la compañía de la gente que aparece en las próximas líneas. Sin duda, habrá personas que hayan aportado su granito de arena y que no aparezcan, mis más sinceras disculpas por adelantado por estos olvidos.

En primer lugar, quiero agradecer todo a mis padres. Sin vuestro apoyo nada de esto hubiese sido posible. Ninguna palabra es suficiente para agradecer todo lo que habéis hecho por mí. Gracias por compartir mis ilusiones y apoyarme en todo momento.

Este proyecto también va dedicado a mis hermanos y a mi cuñado. Al pequeño, por dejarme trabajar e interesarse por lo que estaba haciendo aún cuando lo que más le apetecía era jugar conmigo. A mi hermana y a mi cuñado por su apoyo incondicional y por preocuparse por mí en cada instante.

Gracias también a mis abuelos, a mis primos y a mis tíos, especialmente a tí Pili, por compartir mis sueños y alegría por finalizar la carrera.

La siguiente en la lista eres tú, Patry, la persona que mejor me conoce. Gracias por todo, por aguantarme en los buenos y malos momentos, por saber llevarme, porque sabes que a tu lado todo es más fácil.

No me puedo olvidar de la gente de la residencia, que siempre han estado ahí aguantando mis explicaciones sobre este trabajo. Quiero agradecerlo especialmente a Taoufik y a mis compañeros de piso, Alberto y Fer por todos los buenos momentos que hemos pasado. También a ti, Borja, por amenizarme esas aburridas mañanas de agosto y saber escucharme en los peores momentos.

Quiero dedicar también este proyecto a Pedro y a David, porque aún en la distancia, han aportado algo a su manera. Gracias por ser como sois.

Dentro de la universidad, quiero agradecer en especial a Albert todo lo que ha hecho para que esto fuera posible. Gracias por creer en mí desde un principio y por enseñarme a trabajar de forma correcta. Gracias también a Pablo por esos duros comienzos y a Carlos Jesús por su disposición a ayudarme siempre que lo necesitaba.

Todas las palabras son pocas para agradecer la ayuda recibida en este proyecto, gracias a todos de corazón, y en especial a todos los que me habéis apoyado, vosotros sabéis quienes sois.

Índice general

Resumen **11**

I Introducción **12**

1 Introducción **13**
1.1 Motivación del proyecto 13
1.2 Objetivos del proyecto 14
1.3 Estructura del proyecto 15

II Convergencia de Movilidad y QoS en Redes Móviles **18**

2 Redes Inalámbricas **19**
Historia 19
Componentes de la arquitectura 20
Topología de red 21
Red Ad-Hoc 22
Red infraestructura 22
Estándares 23
Grupos de trabajo IEEE 802.11 25

3 QoS en Redes 802.11 **29**
Capa física 29
Tecnologías basadas en espectro ensanchado 31
Tecnologías basadas en infrarrojos 32
Capa de control de acceso al medio (MAC) 33
Función de Coordinación Distribuida (DCF) 33
Función de Coordinación Puntual (PCF) 37
Extensión QoS: IEEE 802.11e 37
EDCA 38
HCCA 41

4 Movilidad Independiente del Medio **42**
Objetivos 43
Arquitectura 45
Servicios MIH 46
Handover vertical 50

5 Daidalos II **53**
Arquitectura general 54
Arquitectura de QoS 57
Conceptos de la arquitectura 58
Clasificación a nivel de enlace 58
Gestión de los recursos 58
Diseño de la arquitectura e interfaces 60

III **Desarrollo Capa de Abstracción para Nivel Radio** **64**

6 Controlador 802.11e EDCA **65**
Driver Madwifi 65
Parámetros EDCA 68
Modificación a través de ioctl 71

7 Distribución de parámetros en una misma WLAN **75**
Socket 75

8 Interfaz MIHF - RAL **79**
Primitivas MIHF – RAL 79
9.2 Implementación 82

I**V Estudio de prestaciones** **84**

9 Herramientas utilizadas **85**
Aplicaciones para control de interfaces WLAN 85
Wireless-tools 86
iwconfig 86
iwpriv 87
iwlist 88
ifconfig 89
Generación y medida de tráfico 89
Iperf 89
Driver Madwifi 91
Sistema Operativo Linux 92

10 Estudio de prestaciones **93**
Validación de la modificación y distribución de los parámetros EDCA 93
Comunicación entre los módulos MIHF y RAL 95
Validación del conjunto 96

V Conclusiones y trabajos futuros **98**

11 Conclusiones y trabajos futuros **99**
Conclusiones 99
Trabajos futuros 101

VI Apéndices **103**

A Presupuesto **104**
A.1 Introducción 104
A.2 Descomposición en tareas 104
A.2.1 Actividad A: Documentación y análisis del estado del arte 105
A.2.2 Actividad B: Análisis, modificación y validación de los parámetros EDCA 108
A.2.3 Actividad C: Distribución de los parámetros en una misma WLAN 110
A.2.4 Actividad D: Intercambio de primitivas entre RAL y MIHF 111
A.2.5 Actividad E: Elaboración de la memoria del proyecto 112
A.3 Resumen del proyecto 113
A.4 Costes del proyecto 116

B Lista de acrónimos **118**

Referencias **125**

Índice de figuras

2.1 Ejemplo de red ad-hoc 22
2.2 Ejemplo de red infraestructura 23

3.1 Capa física y capa MAC del protocolo 802.11 30
3.2 Tiempos de espera para una transmisión DCF 35
3.3 Proceso RTS/CTS 36
3.4 Categorías de acceso en EDCA 39
3.5 Tiempos de acceso al medio para una transmisión en EDCA 40

4.1 Arquitectura del protocolo 802.21 45
4.2 Representación de los servicios MIH 50
4.3 Ejemplo de handover entre redes heterogéneas 51

5.1 Arquitectura general de un nodo móvil 55
5.2 Arquitectura QoS Daidalos 60
5.3 Arquitectura QoS en WLAN 61
5.4 Esquema general de la funcionalidad a implementar 63

6.1 Estructura del código de Madwifi 67
6.2 Jerarquía de la estructura wmeParams 69
6.3 Funcionamiento de la función ioctl (). 72

8.1 Esquema de primitivas entre MIHF y RAL en un AP 81

10.1 Escenario de validación 93
10.2 Intercambio de mensajes entre MIHF y RAL 96
10.3 Validación del conjunto 97

Índice de cuadros

2.1 Resumen de las tecnologías inalámbricas existentes 25

6.1 Parámetros por defecto en el modo 802.11b 70

8.1 Interfaz A22_AR_MIH_Function::A21_AR_RAL_WLAN . . 80

9.1 Número AC correspondiente a cada tipo de tráfico 88

A.1 Duración, esfuerzo, y horas totales de la Actividad A 114
A.2 Duración, esfuerzo, y horas totales de la Actividad B 114
A.3 Duración, esfuerzo, y horas totales de la Actividad C 115
A.4 Duración, esfuerzo, y horas totales de la Actividad D 115
A.5 Duración, esfuerzo, y horas totales de la Actividad E 115
A.6 Horas de trabajo totales del proyecto 116
A.7 Evaluación de los costes de personal 116
A.8 Evaluación de los costes de material 116
A.9 Evaluación del coste total del proyecto 117

Resumen

En la actualidad están apareciendo en el mercado un gran número de terminales que implementan varias interfaces de diferentes tecnologías radio, tales como 3G y WLAN, entre otras. Sin embargo, el escenario de las telecomunicaciones presenta hoy en día una topología muy compleja a la vista del usuario. Resulta interesante pensar en una nueva infraestructura más flexible, en la que no existan fronteras entre redes heterogéneas, aprovechando las ventajas y características de cada red que se encuentre disponible en la zona de cobertura.

A través de esta idea nace el proyecto europeo Daidalos, que persigue la mejora de las comunicaciones a través de la implementación de un sistema que permita el traspaso entre redes heterogéneas asegurando la continuidad del servicio. Para ello, se basa en el nuevo estándar IEEE 802.21. Aparte de este objetivo principal, se busca el desarrollo general de las comunicaciones, tomando especial importancia la provisión de QoS que, en la actualidad, con el auge de las comunicaciones en tiempo real se convierte en un aspecto fundamental de cualquier infraestructura moderna.

Dentro de la arquitectura Daidalos, se encuentra una entidad denominada RAL (Radio Access Layer), que proporciona la abstracción necesaria para una movilidad independiente del medio. De esta forma, las capas superiores no necesitan conocer las particularidades de cada tecnología específica. Los módulos RAL disponen de funcionalidades de movilidad y de QoS. Dentro de este escenario se enmarca el presente proyecto, teniendo como principal objetivo la implementación de la parte de QoS de la capa RAL de las redes inalámbricas de área local.

Las funcionalidades que este proyecto implementa en la subcapa RAL_WLAN son las siguientes: por un lado, el AP debe ser capaz de distribuir y configurar un conjunto de parámetros para ofrecer QoS que le sean proporcionados. Por otra parte, las estaciones deben de ser capaces de recibir este conjunto de parámetros y configurarlos en sus máquinas.

Parte I

Introducción

Capítulo 1

Introducción

1.1 Motivación del Proyecto

La movilidad se ha convertido en un aspecto indispensable en nuestros días. Prueba de ello es el auge vivido por las redes inalámbricas gracias al aumento de los terminales móviles. Cada vez más, estos dispositivos presentan varias interfaces para diferentes tecnologías de acceso, como 3G o WLAN. Esta nueva característica plantea la necesidad de homogeneizar el acceso al medio, de manera que se abstraiga al usuario de las especificaciones de cada tecnología.

Con este objetivo, nace el proyecto Daidalos, basado principalmente en el estándar IEEE 802.21 que propone una capa de abstracción denominada MIHF (Media Independent Handover Services) que permita realizar handovers entre redes heterogéneas sin perder la continuidad del servicio. Para ello traduce las primitivas independientes del medio en las primitivas específicas de cada tecnología a través del módulo RAL, interfaz entre MIHF y el driver de cada medio.

Cada módulo RAL de cada tecnología específica debe proporcionar funcionalidad de movilidad y de QoS. El presente proyecto trata de implementar la parte de QoS de la capa RAL de las redes inalámbricas de área local.

1.2 Objetivos del Proyecto

Se pueden resumir los objetivos principales del proyecto en los siguientes puntos:

- Estudio del estado del arte de las redes inalámbricas IEEE 802.11, de sus distintas variantes y de las diversas tecnologías que compiten en el mercado junto con IEEE 802.11.

- Estudio del estado del arte actual del protocolo IEEE 802.11e y sus diferentes mecanismos para la provisión de QoS en redes WLAN.

- Estudio exhaustivo de la arquitectura que propone el proyecto europeo Daidalos y de los estándares que utiliza, en particular IEEE 802.21.

- Análisis y especificación de las posibles herramientas a utilizar para realizar el trabajo de investigación.

- Análisis y modificación del controlador inicial (driver Madwifi) para incorporar las funcionalidades requeridas.

- Análisis e implementación de ioctls para la modificación de los parámetros EDCA.

- Análisis e implementación de sockets para la distribución de los parámetros.

- Análisis, especificación e implementación de las primitivas necesarias para la comunicación entre los módulos RAL y MIHF en la provisión de QoS.

- Validación de los resultados obtenidos.

- Extracción de conclusiones y definición de posibles líneas de trabajo para el futuro.

1.3 Estructura del Proyecto

La memoria del presente proyecto está estructurada como se expone a continuación:

- **Parte I: Introducción**

  Introducción general del proyecto, presentación de los objetivos buscados y descripción breve sobre la estructura de la memoria.

- **Parte II: Convergencia de Movilidad y QoS en redes móviles**

  Se presenta el estado actual de las tecnologías en el ámbito en el que se desarrolla el proyecto, en este caso redes inalámbricas LAN. Esta parte se divide en diferentes capítulos:

  - *Capítulo 2. Redes Inalámbricas.* Este capítulo muestra una perspectiva actual de las redes inalámbricas LAN, las tecnologías existentes y su expansión en los últimos años.

  - *Capítulo 3. QoS en Redes 802.11.* Estudio sobre el estándar 802.11 mostrando su capa física y de acceso al medio. Se analiza especialmente la extensión de calidad de servicio, con especial atención a la modalidad EDCA.

  - *Capítulo 4. Movilidad Independiente del Medio.* Estudio del estándar 802.21 que introduce el concepto de handover vertical. Este proceso consiste en el traspaso entre redes heterogéneas, asegurando la continuidad del servicio. Para ello propone una capa independiente del medio, MIHF, que abstraiga a las capas superiores sobre las particularidades de cada tecnología.

  - *Capítulo 5. Daidalos II.* Estudio de la arquitectura que propone este proyecto europeo. Se basa en el estándar 802.21, con el objetivo de mejorar la

movilidad entre redes heterogéneas. Se presta especial atención al módulo RAL, al ser importante para este proyecto.

- **Parte III: Desarrollo de una Capa de Abstracción para el Nivel Radio**

  En esta parte se expondrá la implementación de la subcapa RAL.

  - *Capítulo 6. Controlador 802.11e EDCA.* Este capítulo muestra los pasos dados para la configuración de un conjunto de parámetros EDCA en el driver a través de una ioctl.

  - *Capítulo 7. Distribución de parámetros en una misma WLAN.* Se detalla la herramienta necesaria, un socket a nivel 2, para la distribución de un conjunto de parámetros EDCA en una red WLAN.

  - *Capítulo 8. Interfaz MIHF-RAL.* Se describen las primitivas necesarias para la comunicación entre estos dos módulos que componen la arquitectura Daidalos II.

- **Parte IV: Estudio de las prestaciones**

  Se mostrarán las herramientas necesarias para el desarrollo y las pruebas realizadas al módulo implementado. Esta parte la componen los siguientes capítulos:

  - *Capítulo 9. Herramientas utilizadas.* Se detallan las herramientas utilizadas para el desarrollo del proyecto. Es necesaria una correcta configuración de la red, la generación y medida del tráfico y la modificación del driver Madwifi.

  - *Capítulo 10. Estudio de las prestaciones.* Se explican las pruebas realizadas al módulo implementado.

- **Parte V: Conclusiones y trabajos futuros**

  Esta parte final muestra las conclusiones alcanzadas tras la realización del proyecto y las posibles líneas futuras a seguir para avanzar en el desarrollo iniciado.

- **Parte VI: Apéndices**

  Los apéndices que anexa este proyecto son:

  - *Apéndice A. Presupuesto.* Se estima el gasto necesario para la realización de este proyecto.

  - *Apéndice B. Lista de acrónimos.* Una lista con los acrónimos utilizados durante la escritura del proyecto, con intención de facilitar la lectura.

Parte II

Convergencia de movilidad y QoS en Redes Móviles

Capítulo 2

Redes Inalámbricas

Desde hace relativamente poco tiempo se está produciendo una revolución en el acceso a Internet tal y como lo conocíamos. Se está viviendo un auge de las redes inalámbricas gracias al aumento de terminales móviles y al abaratamiento de la tecnología. Pese a ofrecer menor velocidad y fiabilidad que la transmisión por cable, las redes inalámbricas se han convertido en serias competidoras de las redes cableadas debido a las grandes ventajas que aportan, entre las que destacan la movilidad y libertad de ubicación de los terminales.

A continuación se describe cómo surgieron las redes inalámbricas. Aunque su éxito comercial sea reciente, fueron diseñadas hace varias décadas.

2.1 Historia

Las redes inalámbricas no han sabido conquistar el mercado hasta la actualidad. Aunque con un gran nivel de aplicabilidad a distintos escenarios donde el cable resulta inadecuado o imposible, la falta de estándares y sus reducidas prestaciones en cuanto a velocidad limitaron tanto el interés de la industria como el de los usuarios durante años.

Su impulso comienza a mediados de los 80s cuando la Comisión Federal de Comunicaciones (FCC), el organismo americano encargado de regular las emisiones radioeléctricas, aprueba el uso civil de la tecnología de transmisiones de espectro ensanchado (SS) en la banda ISM (de aplicaciones industriales,

científicas y médicas). A esta concesión se une el desarrollo de un estándar por parte del Instituto de Ingenieros Eléctricos y Electrónicos (IEEE), el IEEE 802.11-1997, que proporciona el factor necesario de estabilidad e inter-operabilidad imprescindible para su comercialización.

La primera publicación de este estándar definía una tasa binaria de 2 Mbps, que podía reducirse a 1 Mbps en entornos hostiles con la técnica de modulación DSSS (Direct Spread Spectrum Sequence) y una tasa binaria de 1 Mbps con modulación FHSS (Frequency Hopping Spread Spectrum Secuence), pudiendo aumentar la velocidad a 2 Mbps en entornos favorables. El estándar sufrió varias actualizaciones que ofrecían mejoras en ciertas características críticas como la velocidad, llegando a conseguir hasta 11 Mbps en la revisión más extendida hasta el momento, IEEE 802.11b.

A continuación se describen los elementos que componen las redes inalámbricas, las diferentes topologías existentes y los estándares desarrollados.

2.2 Componentes de la Arquitectura

Los elementos básicos que componen una red inalámbrica son los siguientes:

El punto de acceso (**Access Point**, AP): consiste en un dispositivo inalámbrico central de una red inalámbrica WiFi que por medio de ondas de radiofrecuencia recibe información a través de diferentes dispositivos móviles y la transmite a través de cable al servidor de la red cableada. El estándar IEEE 802.11 es bastante ambiguo y no define con claridad todas las funciones que debería realizar este elemento.

Dispositivos móviles, tales como ordenadores portátiles, PDAs y teléfonos celulares capaces de enviar y recibir información del punto de acceso de manera inalámbrica.

Otros elementos, tales como dispositivos fijos con tecnología WiFi y amplificadores y antenas para mejorar la calidad de la señal.

Este conjunto de elementos forman una arquitectura celular en la que un punto de acceso es la estación responsable del control de la celda, mientras que las estaciones móviles son aquellas que usan los usuarios finales para realizar la comunicación de datos. Al conjunto formado por un AP y varias estaciones móviles se le denomina Conjunto Básico de Servicios (BSS).

Lo descrito anteriormente es el esquema más sencillo de una red WLAN. Sin embargo, la arquitectura más común es la formada por varias BSS, de tal manera que los diferentes AP se conectan entre sí utilizando algún tipo de red de área local (Ethernet, WLAN, …). A esta red de interconexión entre APs se le llama Sistema de Distribución (DS). El conjunto formado por las diferentes BSS y DS se denomina Conjunto de Servicios Extendido (ESS) y se comporta como una única red de área local.

2.3 Topología de Red

El estándar IEEE 802.11 define dos modos de funcionamiento o topologías de red, que son las redes Ad-Hoc y las redes en Infraestructura.

2.3.1 Redes Ad-Hoc

Su nombre exacto es Conjunto Básico de Servicios Independientes (IBSS), aunque comúnmente son conocidas como redes ad-hoc o de igual a igual. Esta topología compone la configuración de red más básica de una WLAN. Consiste en dos o más terminales móviles con tecnología WiFi, estableciéndose la comunicación entre los nodos directamente, sin necesidad de un AP. Para que la comunicación sea posible hace falta que cada uno de los terminales se encuentre en el rango de cobertura radioeléctrico del resto.

La principal ventaja de esta topología es su sencillez en la implementación, no siendo necesario ningún tipo de gestión administrativa. Únicamente aparecen los nodos móviles y un protocolo descentralizado (p. ej. CSMA/CA) en el que todos los

nodos tienen la misma funcionalidad. Un ejemplo de red ad-hoc se muestra en la figura 2.1.

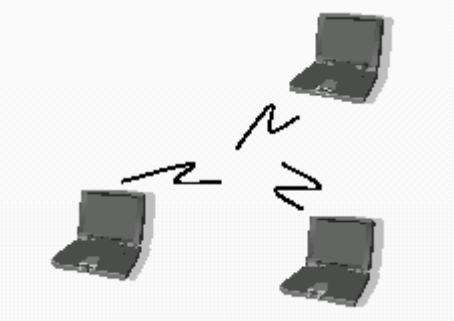

Figura 2.1: Ejemplo de red ad-hoc

2.3.2 Redes en Infraestructura

Esta topología consta de dos elementos, los nodos móviles (MN) y los puntos de acceso (AP).

El AP es el encargado de la gestión y control de las operaciones que realiza un nodo en la red, tales como el ingreso o abandono de un nodo y el control de las prestaciones de las estaciones. Para ello utiliza el envío de paquetes periódicos, llamados paquetes beacon, con información actualizada sobre la red. La zona de cobertura viene delimitada por el AP, ya que todos los nodos deben tener conectividad con este elemento.

Este tipo de red es idóneo para dar servicio a redes permanentes o que ofrecen conectividad a un gran número de nodos, ya que el AP se puede encargar de conectar esta red con otras redes cableadas.

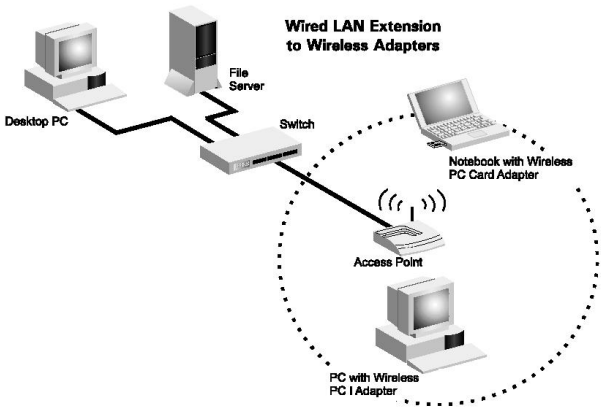

Figura 2.2: Ejemplo de red infraestructura

2.4 Estándares

En este apartado se describen brevemente los principales estándares existentes para la implementación de redes inalámbricas.

HiperLAN 2 (High Perfomance Wireless LAN Type 2)

Estándar europeo desarrollado por la ETSI (Instituto de Estandarización de Telecomunicaciones Europeo) para redes WLAN de alta velocidad (hasta 54 Mbps) en la banda de 5 GHz. Emplea modulación OFDM (Orthogonal Frequency Division Multiplexing). Aún no existen productos comerciales.

IEEE 802.15

Familia de estándares para redes inalámbricas de área personal (PAN). Uno de los más utilizados es el estándar IEEE 802.15.1, variante del conocido Bluetooth.

IEEE 802.16

Familia de estándares para redes inalámbricas de banda ancha metropolitana (MAN), conocido coloquialmente como WiMAX. Utiliza modulación OFDM y consigue tasas para la transmisión de datos de hasta 75 Mbps.

HomeRF – SWAP (Shared Wireless Access Protocol)

Estándar adaptado a las características del hogar. Funciona en la banda ISM de 2.4 GHz, usa como técnica de modulación FHSS, y ofrece una tasa binaria de 2 Mbps. Se utiliza para comunicación inalámbrica de voz y datos a bajo coste.

Bluetooth

Utiliza la banda ISM de 2.4 GHz, técnica de modulación FHSS y ofrece una tasa binaria de 1 Mbps. Esta pensando para la comunicación inalámbrica de voz y datos a corta distancia, conectando diferentes tecnologías tales como equipos informáticos, teléfonos móviles y periféricos.

IEEE 802.11

Estándar para redes de área local inalámbricas creado por IEEE. Se trata de la tecnología más implementada en la actualidad tanto en el ámbito empresarial como en el entorno doméstico y en las redes de acceso público. Esto es debido principalmente a la sencillez del protocolo y a la importancia de ser el estándar desarrollado por una institución tan prestigiosa como IEEE. No obstante, el protocolo presenta muchas deficiencias y aspectos claramente mejorables, por lo que se comenzó a trabajar en diferentes líneas para la mejora del protocolo IEEE 802.11, que serán desarrollados en el siguiente apartado.

Se muestra una tabla resumen con las principales características de cada una de las tecnologías citadas anteriormente.

Tecnología	Banda de Frecuencias (GHz)	Velocidad máxima (Mbps)	Organización
802.11	2.4 – 5	54	IEEE
HiperLAN	5	54	ETSI
Bluetooth	2.4	2	Bluetooth SIG
WiMAX	Menor de 11	70	IEEE 802.16ª
HomeRF	2.4	2	Intel

Cuadro 2.1: Resumen de las tecnologías inalámbricas existentes

2.5 Grupos de trabajo IEEE 802.11

A continuación se detallan los diferentes Grupos de Trabajo (TGs) del IEEE encargados de la mejora de los diferentes aspectos.

IEEE 802.11a

Utiliza frecuencias superiores a los 5 GHz, por lo que no es compatible con el estándar 802.11b y el estándar 802.11g. Permite realizar transmisiones con velocidades máximas de 54 Mbps y utiliza modulación OFDM.

Es útil en situaciones en las que sea necesario separar el tráfico o para zonas con mucho ruido e interferencias debido a que la banda de 2.4 GHz tiene gran uso (teléfonos inalámbricos y hornos microondas entre otros aparatos). Sin embargo, requiere de un mayor número de puntos de acceso al tener menor alcance por trabajar en una frecuencia mayor, con lo que las ondas son absorbidas con mayor facilidad.

IEEE 802.11b

802.11b tiene una velocidad máxima de transmisión de 11 Mbps y utiliza el método de acceso CSMA/CA (Carrier Sense Multiple Access/Collision Avoidance) definido en el estándar original. Funciona en la banda de 2.4 GHz. Debido al espacio ocupado por la codificación del protocolo CSMA/CA, en la práctica, la velocidad

máxima de transmisión con este estándar es de aproximadamente 5.9 Mbps sobre TCP y 7.1 Mbps sobre UDP.

Fue la extensión con más éxito comercial y actualmente comparte ese éxito con IEEE 802.11g.

IEEE 802.11d

Su objetivo es definir una capa física que pueda operar en el mayor número posible de países. Sus líneas de trabajo persiguen definir una nueva funcionalidad a nivel MAC, de manera que los AP reciban información acerca de los niveles máximos de potencia y los canales permitidos en cada país.

IEEE 802.11e

Con el estándar 802.11e, la tecnología IEEE 802.11 soporta tráfico en tiempo real gracias a las garantías de Calidad de Servicio (QoS) incorporadas. El objetivo del nuevo estándar 802.11e es introducir nuevos mecanismos a nivel de capa MAC para soportar dichos servicios. Para cumplir con este objetivo, introduce un nuevo elemento llamado Hybrid Coordination Function (HCF) con dos tipos de acceso: EDCA y HCCA.

Debido a la relevancia que toma este estándar en el presente proyecto, será desarrollado con detalle en un capítulo posterior.

IEEE 802.11f

Trabaja en la estandarización de un protocolo único que permita la itinerancia de estaciones entre AP de diferentes compañías. El protocolo resultante es el IAPP (Inter-Access Point Protocol).

IEEE 802.11g

El TGg es el encargado de mejorar la tasa de transferencia de 802.11b en la banda de ISM. Para ello utiliza como modulación OFDM, que permite obtener velocidades de hasta 54 Mbps. Es compatible con 802.11b. En la actualidad es, junto con 802.11b, el estándar de mayor éxito comercial. La mayoría de los dispositivos wireless dan soporte IEEE 802.11b/g.

IEEE 802.11h

Se trata de una versión europea del IEEE 802.11a. El TGh centra su trabajo en la gestión del espectro de 802.11a, incluyendo mejoras como la selección dinámica de canales y el control de potencia transmitida para evitar interferencias.

IEEE 802.11i

El TGi tiene como objetivo la mejora de la seguridad del estándar IEEE 802.11, punto débil de este estándar. Define el uso de protocolos criptográficos para mejorar la seguridad en las redes inalámbricas de área local.

IEEE 802.11j

El objetivo del TGj es ampliar el estándar 802.11 para que pueda operar en la banda japonesa de 4.9-5 GHz. Busca cumplir las especificaciones del organismo regulatorio japonés en cuanto a tasa de transferencia, potencia radiada, detección de canal, emisiones espúreas y modos de operación.

IEEE 802.11k

El trabajo del TGk persigue la inclusión de nuevas variables en la MIB (Management Information Base) de la capa MAC, que permitirá la gestión remota de la red WLAN a partir de medidas de la capa física y MAC.

IEEE 802.11n

El grupo TGn tiene como objetivo la mejora de los actuales estándares, prometiendo velocidades reales de transmisión de hasta 600 Mbps. También se espera que el alcance de operación de las redes sea mayor con este nuevo estándar gracias a la tecnología MIMO (Multiple Input – Multiple Output), que permite utilizar varios canales a la vez para enviar y recibir datos gracias a la incorporación de varias antenas.

Capítulo 3

QoS en Redes 802.11

El estándar IEEE 802.11 define cómo debe ser la capa física y el nivel de enlace de una red inalámbrica de área local. En este capítulo se explica con detalle tanto la capa física como los dos tipos de acceso al medio que existen en este protocolo, PCF y DCF. También se trata con profundidad la extensión de calidad de servicio, IEEE 802.11e, centrándose en el modo de acceso EDCA por ser el más implementado comercialmente.

En el próximo apartado se describe el protocolo IEEE 802.11 desde el nivel más bajo, es decir, desde la capa física, de la que se explica las subcapas que la componen y sus funciones junto con las tecnologías que utiliza como medio físico. A continuación, se detalla la capa de control de acceso al medio (MAC), donde destaca el modo de acceso basado en contienda DCF (Distributed Coordination Function) por ser el modo de acceso más común y el utilizado por los mecanismos de Calidad de Servicio. Existe otro modo de acceso al medio mediante control centralizado, PCF (Point Coordination Function), con implementación escasa en el mercado, por lo que se describirá brevemente.

3.1 Capa física

La capa física (PHY: Physical Layer) se encarga de las gestiones necesarias para realizar correctamente la comunicación usando el medio físico (p. ej. Codificación de canal, niveles de potencia, modulación, etc), proporcionando una serie de servicios a la capa MAC o capa de acceso al medio. Los servicios que ofrece la capa física permiten que la capa de nivel superior no tenga que

preocuparse de la capacidad del medio físico o de cómo transmitir a través de él, pues de esas funciones ya se encarga esta capa.

La capa física se divide en dos subcapas:

- **Subcapa Dependiente del Medio (PMD)**: define las características y la manera de transmitir y recibir a través de un medio sin cables entre dos o más estaciones.

- **Procedimiento de Convergencia de la Capa Física (PLCP)**: define una forma de traducir MPDUs o unidades de datos MAC en un formato de tramas susceptibles de ser transmitidas o recibidas entre diferentes estaciones a través de la capa PMD.

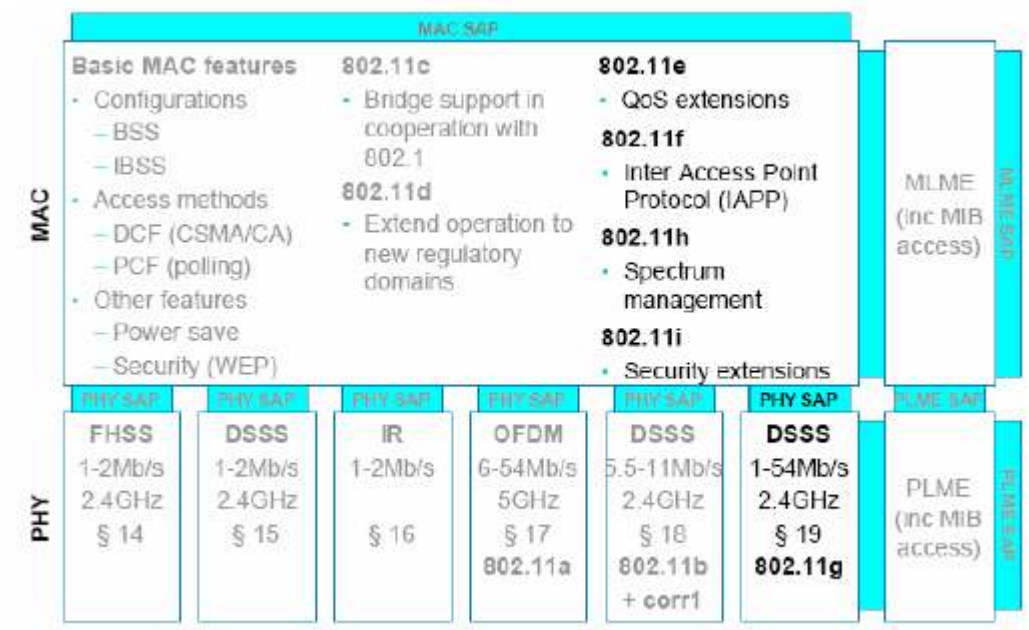

Figura 3.1: Capa física y capa MAC del protocolo 802.11

La comunicación entre MACs de diferentes estaciones se realiza usando la capa física mediante una serie de puntos de acceso al servicio (PHY SAP), donde la capa MAC invoca las primitivas de servicio correspondientes.

Además de estas capas, podemos distinguir la capa de gestión. Esta capa está formada por la subcapa MAC de gestión y la subcapa física de gestión. En esta capa se encuentra la estructura MIB (Management Information Base) que contiene las variables de

gestión, los atributos, las acciones y las notificaciones requeridas para gestionar una estación.

Inicialmente, 802.11 define 3 capas físicas diferentes, 2 de espectro ensanchado y una de infrarrojos. A continuación se explican con más detalle estas tres tecnologías.

3.1.1 Tecnologías basadas en espectro ensanchado

La tecnología de espectro ensanchado utiliza todo el ancho de banda disponible en lugar de usar una portadora para concentrar la energía en una frecuencia. La banda se comparte entre todos los usuarios que se comunican en dicho rango, siempre que utilicen la misma técnica de espectro ensanchado. Entre las ventajas que aporta se encuentran una mayor inmunidad a las interferencias y más posibilidades de encriptación. Existen dos tipos de tecnología de espectro ensanchado:

Espectro Ensanchado por Secuencia Directa (DSSS)

En esta técnica se genera un patrón de bits redundantes (señal de chip) para cada uno de los bits que componen la señal. Cuanto mayor sea esta señal, mayor será la resistencia a las interferencias. En recepción es necesario realizar el proceso inverso para obtener la información original.

Una vez aplicada la señal de chip, el estándar IEEE 802.11 ha definido dos tipos de modulación para la técnica de espectro ensanchado por secuencia directa, la modulación DBPSK (Differential Binary Phase Shift Keying) y la modulación DQPSK (Differential Quadrature Phase Shift Keying), que proporcionan una velocidad de transferencia de 1 Mbps y 2 Mpbs respectivamente.

Mediante la revisión IEEE 802.11b se consiguen velocidades de transmisión de hasta 11 Mbps, lo que incrementa notablemente el rendimiento de este tipo de redes.

Espectro ensanchado por salto en frecuencia (FHSS)

La tecnología de espectro ensanchado por salto en frecuencia (FHSS) consiste en transmitir una parte de la información en una determinada frecuencia durante un intervalo de tiempo. Pasado este tiempo se cambia la frecuencia de emisión y se sigue transmitiendo a otra frecuencia. De esta manera cada tramo de información se transmite en una frecuencia distinta durante un intervalo muy corto de tiempo. El orden en los saltos en frecuencia se determina según una secuencia pseudoaleatoria almacenada en unas tablas, y que tanto el emisor como el receptor deben conocer. Si se mantiene la sincronización en los saltos en frecuencia se consigue que, aunque en el tiempo se cambie de canal físico, a nivel lógico se mantenga un único canal por el que se realiza la comunicación.

El estándar IEEE 802.11 define la modulación aplicable en este caso. Se utiliza la modulación en frecuencia FSK (Frequency Shift Keying), con una velocidad de transmisión de 1 Mbps, ampliable a 2 Mbps. En la revisión del estándar, IEEE 802.11b también se ha ampliado esta velocidad a 11 Mbps.

La ventaja de FHSS frente DSSS es que pueden existir varios AP transmitiendo en la misma zona sin que existan interferencias entre ellos, siempre que no coincidan transmitiendo en el mismo instante de tiempo y usando portadoras de la misma frecuencia.

3.1.2 Tecnologías basadas en infrarrojos

Las WLAN por infrarrojos son aquellas que usan el rango infrarrojo del espectro electromagnético para transmitir información mediante ondas por el espacio libre. Los sistemas de infrarrojos se sitúan en altas frecuencias, justo por debajo del rango de frecuencias de la luz visible. Las propiedades de los infrarrojos son, por tanto, muy similares a las de la luz visible, por lo que estas ondas son susceptibles de ser interrumpidas por cuerpos opacos necesitando visión directa para la comunicación.

Esta tecnología no se ha implementado en la práctica debido a dos inconvenientes claramente reconocibles: el medio físico de

infrarrojos presenta una alta atenuación expuesto a la luz solar, de manera que hace viable su uso únicamente en entornos cerrados. Por otra parte, como se ha comentado anteriormente, es necesaria visión directa entre los terminales. Esto, unido al buen funcionamiento de las técnicas de espectro ensanchado, ha hecho que la tecnología infrarroja quede obsoleta.

3.2 Capa de control de acceso al medio (MAC)

La norma IEEE 802.11 define una única capa MAC para todas las redes físicas, ayudando a la fabricación en serie de chips.

Se definen dos tipos de funciones de acceso al medio:

- **Función de Coordinación Distribuida (DCF)**: emplea un protocolo de contienda (CSMA/CA).

- **Función de Coordinación Puntual (PCF):** emplea un protocolo de control centralizado, en el que hay un punto de coordinación que suele ser el AP, que selecciona quién puede transmitir en cada momento. Este sistema sólo funciona en arquitecturas de red en infraestructura.

3.2.1 Función de Coordinación Distribuida (DCF)

La Función de Coordinación Distribuida es el protocolo MAC básico de 802.11. Implementa CSMA/CA (Carrier Sense Multiple Access / Collision Avoidance), un sistema de acceso al medio que consiste en escuchar antes de transmitir. Se trata de una técnica basada en contienda, por lo que el tráfico que se transmite presenta retardos no deterministas.

A continuación se explica con detalle el funcionamiento del sistema CSMA y CA.

CSMA

CSMA es un protocolo que permite que múltiples estaciones utilicen un mismo medio de transmisión. Mediante el uso de este sistema, una estación determina si el medio está ocupado o no y, en función de ello, si puede transmitir o debe esperar. Por lo tanto, la decisión de acceder en un instante de tiempo al medio está repartida entre todas las estaciones.

Esta técnica presenta un punto débil, las colisiones. Este problema se produce cuando dos estaciones escuchan el medio y detectan que está libre en el mismo instante de tiempo. Al transmitir las dos estaciones a la vez, se produce una colisión, lo que invalida la transmisión para, al menos, una de las dos estaciones. Esto produce una pérdida de eficiencia al tener que reenviar la información colisionada.

CA

La solución propuesta para el problema de las colisiones es el sistema de prevención de colisiones CA (Collision Avoidance). Consiste en que la estación que recibe un paquete correctamente, responda al remitente con un paquete de asentimiento (ACK). Si la estación transmisora no recibe un asentimiento, asume que el paquete no ha sido recibido correctamente y pasa a reenviar dicho paquete.

Antes de transmitir cualquier tipo de trama, existen unos tiempos de espera que el transmisor utiliza para comprobar que el medio está libre. A continuación se explica el proceso con más detalle.

El modo DCF actúa de la siguiente manera. Cuando una estación quiere transmitir un paquete de datos, inicia un proceso conocido como backoff. El proceso de backoff consiste en escuchar si el medio está libre durante un tiempo DIFS (DCF Interframe Space). Con el objetivo de disminuir el número de colisiones, se añade un periodo de tiempo aleatorio adicional llamado backoff time.

El backoff time se obtiene de manera independiente y aleatoria en cada estación y para cada paquete que se quiera enviar. Este

tiempo es múltiplo de un periodo de tiempo fijo y el número de periodos que la estación espera se elige aleatoriamente entre cero y el tamaño de la ventana de contención (CW, Contention Window), con una distribución uniforme. La primera vez que se intenta enviar un paquete, CW coincide con CWmin, por lo que aunque se elija el valor aleatoriamente, todas las estaciones presentan las mismas probabilidades de acceso al medio.

Para reducir la probabilidad de colisión, después de cada intento de transmisión fallido, la ventana de contención doblará su tamaño hasta un máximo establecido (CWmax). Por otro lado, cada vez que un paquete es transmitido correctamente la ventana de contención pasara a tener el tamaño mínimo permitido (CWmin), con lo que se mejora el rendimiento del canal.

Si la transmisión se ha realizado con éxito y no hay más paquetes que transmitir, la estación inicia otro backoff time aleatorio, conocido como post-backoff. Este mecanismo asegura que siempre exista un tiempo aleatorio entre dos paquetes consecutivos.

Solo hay una excepción al proceso de backoff. Si llega un paquete para ser transmitido y se cumplen las siguientes condiciones: la cola de transmisión está vacía, el último post-backoff ha sido completado y el medio ha estado vacío durante el último DIFS.

Para la transmisión del paquete ACK, la estación receptora espera un tiempo SIFS (Short Inter Frame Space) después de la recepción del paquete. Un tiempo SIFS es menor que un tiempo DIFS, con lo que se asegura que dos estaciones terminan su dialogo antes de que otra estación intervenga en el medio.

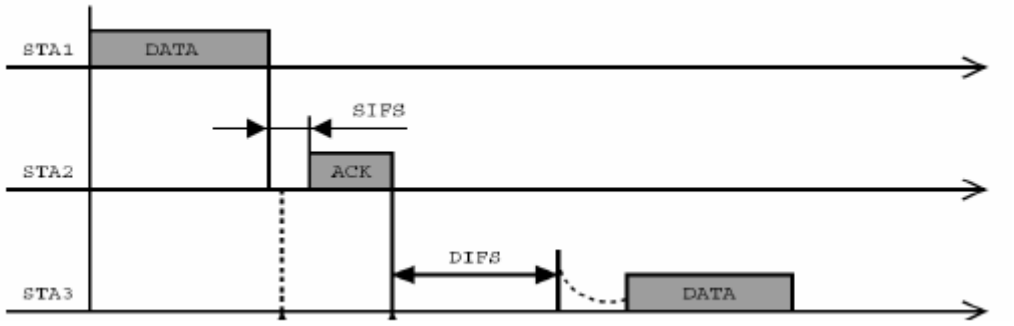

Figura 3.2: Tiempos de espera para una transmisión DCF

Sin embargo, el sistema hasta ahora descrito tiene un problema: estamos suponiendo que una estación es capaz de saber si el medio está ocupado o no, simplemente escuchando. Esto, en redes inalámbricas centralizadas es imposible, ya que una estación no tiene por que estar en el radio de cobertura de otra estación. A esta situación se la denomina el problema de los nodos ocultos.
Si una estación A no puede detectar si una estación B está transmitiendo o no, el mecanismo de CSMA/CA degenera en un simple ALOHA, donde los terminales transmiten cuando lo necesitan, sin importar si hay una transmisión en progreso. Para evitar esto se complementa el CSMA/CA con el envío de paquetes RTS/CTS.

RTS/CTS

El sistema de RTS/CTS consiste en, antes de enviar los datos, calcular el tiempo total que estará el canal ocupado por esta transacción y enviarlo al receptor en una trama corta, denominada RTS (Request To Send). El receptor de un RTS lo duplica y lo manda en un CTS (Clear To Send) a broadcast. De esta manera todos los nodos de la BSS susceptibles de interferir con la comunicación, están al tanto de la duración de la misma. Cuando un nodo recibe un CTS, almacena la información de cuanto va a estar el canal ocupado en un NAV (Network Allocation Vector). La función de este vector es indicar el tiempo previsto de uso del canal, de esta manera una estación puede saber cuando el canal estará libre.

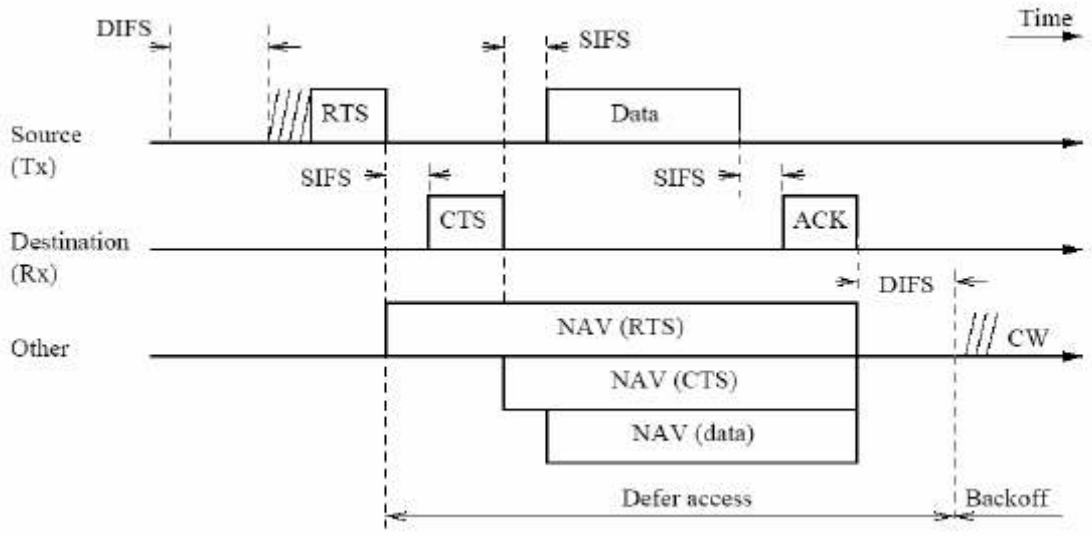

Figura 3.3: Proceso RTS/CTS

Mientras el NAV en el nodo receptor del CTS le indique que no debe transmitir, permanecerá callado, incluso si aparentemente, para él, el canal está libre. De esta manera la posibilidad de colisión por nodo oculto se ve disminuida enormemente, ya que ahora solo pueden colisionar los RTS/CTS, que son tramas de duración muy corta.

3.2.2 Función de Coordinación Puntual (PCF)

La Función de Coordinación Puntual es una técnica que controla el acceso al medio de manera centralizada a través de un elemento principal de la red que otorga permisos para transmitir. Este elemento suele ser el AP. Este método de acceso fue creado para ofrecer calidad de servicio, ya que centralizando las comunicaciones se puede dar mayor prioridad a un transmisor frente a otro.

Sin embargo, presenta problemas en su definición que provocan que no se pueda asegurar la calidad de servicio, por lo que su implementación ha sido escasa, siendo utilizada de forma generalizada la técnica de acceso al medio DCF.

Para ampliar las posibilidades de este protocolo en cuanto a calidad de servicio, el IEEE desarrolló una extensión denominada 802.11e que será detallada a continuación.

3.3 Extensión de QoS: IEEE 802.11e

En los últimos años se ha producido un auge del contenido multimedia y de las aplicaciones en tiempo real. A la hora de transmitir este tipo de información surgió la necesidad de dotar a las redes de unas herramientas básicas que permitieran proporcionar calidad de servicio, de forma que se asegurasen los requerimientos exigidos para este tipo de comunicaciones.

Este hecho, junto a la expansión de las redes inalámbricas, llevó al grupo de estandarización IEEE al desarrollo de una nueva extensión que dotase de soporte para ofrecer calidad de servicio a las redes 802.11 a través de la mejora de la capa MAC, lo que dio lugar al nacimiento del estándar 802.11e.

El nuevo estándar define el protocolo HCF (Hybrid Coordination Function) para obtener el objetivo descrito, introduciéndose dos mecanismos de acceso al medio:

- **EDCA** (Enhanced Distributed Channel Access): extensión del protocolo DCF de las redes 802.11.
- **HCCA** (HCF Controlled Channel Access): control de acceso al medio de forma centralizada (extensión del protocolo PCF de las redes 802.11).

Comercialmente ha tenido más éxito EDCA, por lo que en el presente proyecto se utilizará un controlador basado en este mecanismo.

A continuación se muestra con mayor detalle el funcionamiento de EDCA. Posteriormente se explicará brevemente el protocolo HCCA, no ahondando más en él por su escasa aceptación comercial.

3.3.1 EDCA

Su funcionamiento se basa en dos nuevos conceptos: las categorías de acceso (ACs) y múltiples entidades independientes de backoff. En cada estación hay 4 ACs asociadas cada una de ellas a una entidad backoff independiente.

Cada AC se utiliza para un tipo de tráfico específico. Los 4 tipos de tráfico y su correspondiente AC son de mayor a menor prioridad: voz (AC_VO), vídeo (AC_VI), Best Effort (AC_BE), background (AC_BK). Para poder clasificar los paquetes es necesario que el sistema identifique el tráfico al que pertenecen. La nueva versión de IP, IPv6, lleva en la cabecera información específica sobre el tipo de tráfico de cada paquete.

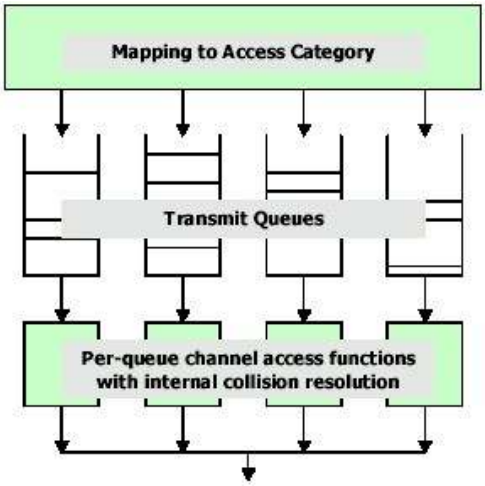

Figura 3.4: Categorías de acceso en EDCA

El acceso al medio es ajustado para cada entidad de backoff mediante unos valores específicos de EDCA. Para el correcto funcionamiento del sistema es necesario que las entidades definidas para un tipo de tráfico tengan los mismos parámetros en todos los nodos de la red. Estos parámetros son cuatro y se describen a continuación:

AIFS

Este parámetro indica el tiempo que debe escuchar una estación el medio libre antes de empezar el proceso de backoff para transmitir una trama.

AIFS es útil para la provisión de QoS debido a que una estación con menor valor en este parámetro, por simple estadística, tiene prioridad en el acceso al canal.

CWmin

Para el primer periodo de backoff se selecciona el tiempo a esperar tras el AIFS según una distribución uniforme entre 0 y CWmin.

Al igual que en el caso anterior, un menor valor de CWmin ofrece mayor prioridad en el acceso al canal.

CWmax

El valor de la ventana de contención crece tras cada transmisión errónea multiplicándose por dos desde el valor de CWmin hasta el de CWmax, por lo que este parámetro define el valor máximo hasta el que puede crecer la ventana de contención.

De la misma forma que en los dos casos anteriores, un menor valor de CWmax ofrece mayor prioridad en el acceso al canal.

TXOP

Este parámetro define el tiempo máximo que una estación puede transmitir tramas tras una transmisión exitosa. Su uso prioriza un tráfico frente a otro, por lo que provee QoS.

Existe la posibilidad de que se produzca una colisión virtual entre dos o más entidades de backoff al intentar transmitir a la vez ya que hay una componente aleatoria. Si se produce este hecho la cola con mayor prioridad transmitiría, actuando el resto como si se hubiera producido una colisión.

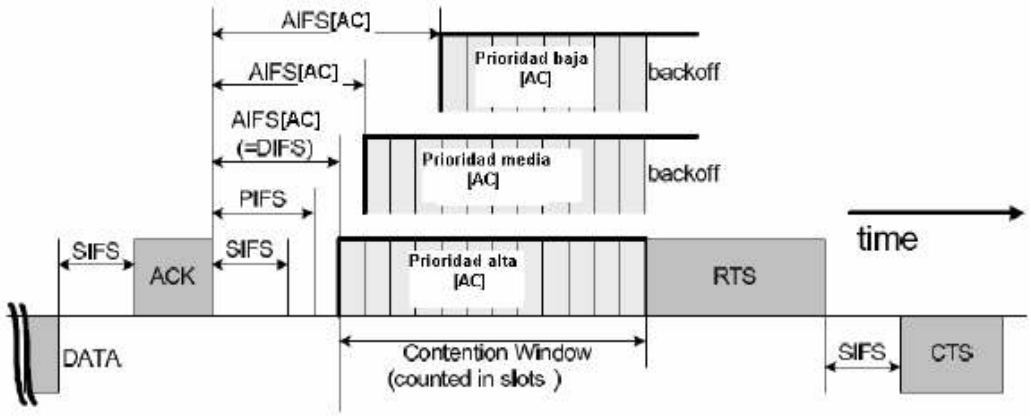

Figura 3.5: Tiempos de acceso al medio para una transmisión en EDCA

3.3.2 HCCA

En este modo de acceso existe una entidad principal llamada Hybrid Coordination (HC), que suele ser el AP, encargada de controlar el acceso del resto de las estaciones. Se indica la estación que debe transmitir mediante paquetes conocidos como QoS CF-Poll, donde se define también el valor del TXOP de manera que se controla el tiempo máximo que puede transmitir. Puesto que el HC debe controlar el acceso al medio de toda la red, debe tener prioridad frente al resto de estaciones. Para ello se le otorga un valor de AIFS menor en todas sus colas.

Este modo no existe de forma independiente al modo EDCA. Se alternan periodos en el que el control del acceso se hace de forma centralizada con otros periodos en los que las estaciones compiten por el medio. Debido a su escasa aceptación comercial, no será tenido en cuenta en el presente proyecto.

Capítulo 4

Movilidad Independiente del Medio

En los últimos años están apareciendo terminales móviles que soportan varias tecnologías, tales como teléfonos móviles con conectividad WiFi. Esta unión plantea nuevos retos y promete una mejora en la conectividad ofrecida al usuario, por lo que el organismo IEEE está desarrollando un estándar, el 802.21, capaz de realizar traspasos (handovers) entre redes heterogéneas asegurando la continuidad, conocido como MIH (Media Independent Handover services).

La funcionalidad que persigue el nuevo estándar IEEE 802.21 MIH consiste en que los terminales sean capaces de escanear las redes inalámbricas disponibles, tanto 3G como WLAN, y se conecten a aquella que le ofrezca un mejor servicio en cada momento.

Para ello, el principal objetivo del protocolo consiste en implementar el traspaso entre redes heterogéneas, llamado handover vertical, asegurando la continuidad de la conexión en este proceso. Las funcionalidades requeridas para proveer la continuidad en la sesión dependen de complejas interacciones específicas para cada tecnología en particular. 802.21 provee una función que permite a las capas superiores acceder a las capas inferiores para proveer la continuidad, sin que éstas tengan que tratar con las especificaciones de cada tecnología en particular. El nuevo protocolo puede ser visto como un “pegamento” entre el sistema IP y los distintos escenarios existentes para redes móviles.

Para conseguir los objetivos descritos, 802.21 define una entidad independiente del medio, como se ha descrito anteriormente, que provee una interfaz genérica entre las diferentes capas de enlace y las capas superiores. Para manejar las

particularidades de cada tecnología, 802.21 traduce esta interfaz genérica en un conjunto de Puntos de Acceso al Servicio dependientes del medio (SAPs) cuyo objetivo consiste en recolectar información y controlar el comportamiento del enlace durante los handovers. Además, se definen también un conjunto de interfaces remotas entre los terminales y la red y entre redes para asistir al terminal en las decisiones de handover.

Todas las funcionalidades que ofrece este nuevo protocolo son proveídas por un conjunto de servicios llamado Eventos, Comandos y Servicios de Información. Estos servicios son el núcleo de la especificación y hacen posible la comunicación entre las capas inferiores y la red.

Este capítulo comienza con la descripción de los principales objetivos perseguidos por este nuevo estándar. A continuación, se detalla la arquitectura y se explica con detalle en qué consiste el conjunto de servicios. Se finaliza con la exposición de un caso práctico de handover, donde se puede apreciar el intercambio de mensajes entre las distintas entidades involucradas.

4.1 Objetivos

Los objetivos perseguidos por el estándar IEEE 802.21 se resumen en los siguientes tres puntos principales:

- Conseguir que el protocolo permita handovers entre redes heterogéneas de forma transparente para el usuario.

- Definición de una nueva capa de enlace SAP que ofrezca una interfaz común independiente de la tecnología específica del medio. Para cada tecnología considerada en 802.21, este SAP es mapeado en las correspondientes primitivas específicas de la tecnología.

- Definición de un conjunto de funciones que permitan la optimización de los handovers.

A pesar de que el principal objetivo de 802.21 sea el permitir handovers entre redes heterogéneas, se ha declarado también un conjunto de objetivos secundarios:

- **Continuidad en el servicio**, definida como la continuación del servicio durante y después del proceso de handover. Se pretende evitar la necesidad de restaurar la sesión por parte del usuario después de un handover.

- **Handover basado en la aplicación**. Diferentes aplicaciones poseen distinta tolerancia al retardo y pérdida de paquetes. Conociendo estas características se puede mejorar el servicio. Por ejemplo, el handover afectaría menos a la comunicación durante la fase de silencio de una conversación de voz.

- **QoS (Quality of Services)**. Es un factor muy importante a la hora de tomar la decisión de handover. Se informa a las capas superiores sobre los niveles exigidos de calidad para no degradar el servicio ofrecido al usuario.

- **Escaneo de redes.** Esencial para proveer nuevas posibilidades al servicio de selección de redes. El estándar 802.21 define las pautas para obtener información relevante sobre las redes vecinas. La información de red suele incluir campos como el tipo de enlace, identificador, calidad, …

- **Selección de red.** Proceso por el cual un nodo móvil o una entidad de una red recolecta información sobre redes vecinas y selecciona un enlace (posiblemente entre muchos disponibles) para ofrecer conectividad al usuario. La selección puede basarse en diversos criterios como la QoS requerida, preferencias del usuario, políticas, etc. Si la red seleccionada no es la actual, será necesario realizar un handover. 802.21 sólo provee las funciones necesarias para asistir la selección de red, no toma la decisión de handover. Esta función es realizada por las capas superiores.

- **Control de la potencia.** El estándar 802.21 ofrece mecanismos que ayudan a alargar la vida de la batería. Por ejemplo, modos de latencia eficientes basados en el estado real del enlace.

4.2 Arquitectura

En esta sección se presenta la arquitectura general de IEEE 802.21. Se describen las diferentes capas que componen la pila del nuevo protocolo y sus interacciones.

En la siguiente figura se presenta un diagrama lógico de la arquitectura general de los diferentes nodos que componen una red 802.21. Se muestra un nodo móvil con una interfaz 802 y otra 3GPP/3GPP2 y el núcleo de estas redes.

Como se puede observar, la arquitectura tiene una estructura común llamada MIHF (Media Independent Handover Function) que actúa como capa intermediaria entre las capas superiores e inferiores, siendo su principal función la de coordinar el intercambio de información y comandos entre los diferentes dispositivos involucrados en el proceso de handover.

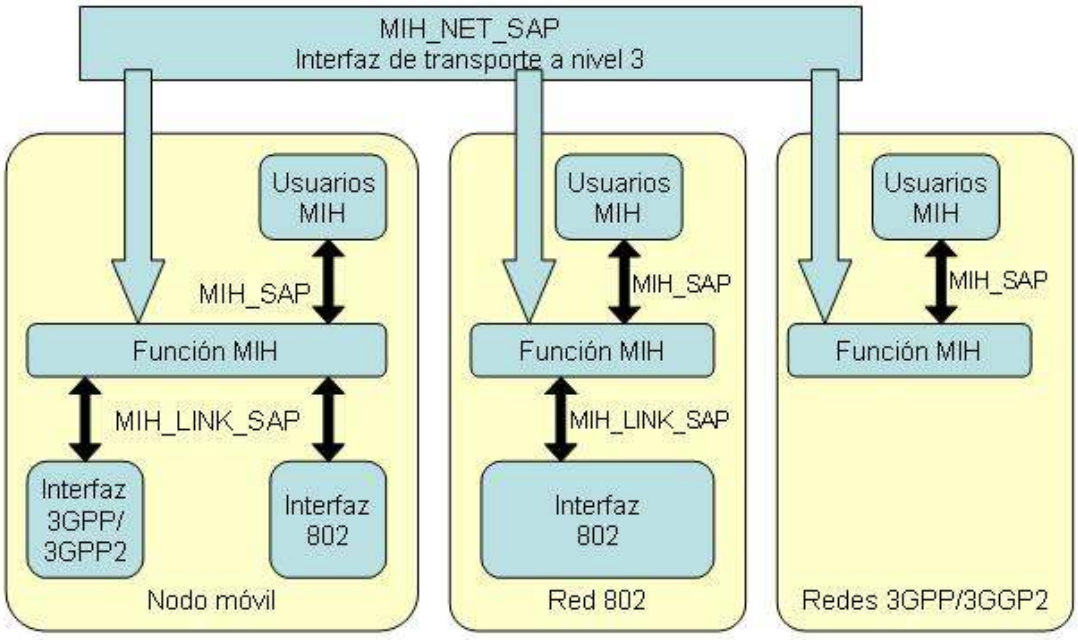

Figura 4.1: Arquitectura del protocolo 802.21

Cada nodo tiene un conjunto de usuarios MIH que suelen ser protocolos de gestión de la movilidad que aprovechan la funcionalidad proporcionada por MIHF para controlar y conseguir información relacionada con el traspaso.

La comunicación entre MIHF y las otras entidades funcionales está basada en un conjunto de primitivas de servicio agrupadas en Puntos de Acceso al Servicio (SAPs). Existen tres categorías de SAPs:

- MIH_SAP: esta interfaz permite la comunicación entre MIHF y la capa superior de usuarios MIH.

- MIH_LINK_SAP: esta interfaz permite la comunicación entre la capa MIHF y las capas inferiores de la pila del protocolo.

- MIH_NET_SAP: permite el intercambio de información entre entidad MIHF remotas.

Todas las comunicaciones entre MIHF y las capas inferiores se realiza a través de MIH_LINK_SAP. Este SAP fue definido como una interfaz independiente del medio, de manera que la capa MIHF pudiese ser diseñada independientemente de las especificaciones de cada tecnología. Sin embargo, estas primitivas son traducidas en las primitivas particulares de cada tecnología considerada en 802.21.

4.3 Servicios MIH

Como se ha podido comprobar, es necesario un continuo intercambio de información entre las diferentes entidades que componen la arquitectura. Para ello, se han definido tres tipos diferentes de comunicación, también llamados servicios MIH: Servicios de Eventos (ES), Servicios de Comandos (CS) y Servicios de Información (IS). Estos servicios permiten que los usuarios MIH accedan a información relativa al handover para enviar comandos a las capas inferiores.

Los servicios MIH pueden ser enviados de forma síncrona o asíncrona. Los eventos generados en las capas de enlace y enviados a MIHF son transmitidos de forma asíncrona, mientras que los comandos y la información, generados por un mecanismo de pregunta/respuesta son enviados de una forma síncrona.

A continuación se detalla cada uno de los servicios MIH.

Servicio de Eventos Independiente del Medio

El estándar IEEE 802.21 soporta que el handover sea iniciado por el terminal móvil o por la red, por lo que los eventos relacionados con los traspasos pueden ser originados en la capa MAC o en la capa MIHF de los nodos o en el punto de acceso a la red.

Como diferentes entidades pueden estar interesadas en los eventos generados, el estándar plantea un mecanismo de subscripción, por el que los eventos serán enviados a la lista de subscriptos.

Los eventos se pueden dividir en dos categorías, Eventos de enlace y Eventos MIH. Los eventos de enlace son aquellos generados en la capa de enlace y con destino la función MIH. Los eventos propagados desde MIHF a los usuarios MIH reciben el nombre de eventos MIH.

Los tipos de eventos generados se pueden clasificar en los siguientes puntos:

- Eventos de cambio de estado en MAC y PHY.
- Eventos de cambio de Parámetros de enlace.
- Eventos de sincronización de enlace. Envían información relevante a las capas superiores sobre las actividades de la capa de enlace.
- Eventos de transmisión de enlace. Estos eventos informan del estado de transmisión a las capas superiores. Estos datos pueden ser usados, entre otras aplicaciones, para dimensionar los buffers necesarios para realizar el handover transparente.

El Servicio de Eventos es útil para detectar cuando es posible realizar un traspaso de red. Hay varios eventos, tales como “Link Up”, “Link Down” y “Link Parameters Change” que pueden ser utilizados para detectar cuando está disponible un nuevo enlace o

cuando las condiciones del canal son apropiadas para llevar a cabo un traspaso a una nueva red.

Servicio de Comandos Independiente del Medio

El Servicio de Comandos Independiente del Medio (MICS) consta de los comandos enviados desde las capas superiores a las capas inferiores con intención de determinar el estado de los enlaces o controlar y configurar el terminal para optimizar las decisiones de traspaso.

Los protocolos de gestión de la movilidad tienen que combinar la información dinámica resultado de monitorizar el estado del enlace y sus parámetros, obtenida de MICS, con la información estática remitida por el estado de la red obtenida del servicio de eventos. A partir de esta información, debe ayudar a la decisión de realizar el handover.

Los comandos se clasifican en dos categorías:

- Comandos MIH: estos comandos son enviados desde las capas superiores a MIHF. En caso de que el comando quiera ser destinado a un MIHF remoto, se envía al MIHF local que lo reenviará a la dirección correcta a través del protocolo de transporte de MIHF. Por otro lado, para permitir handovers iniciados por la red, el servicio de comandos proporciona un conjunto de comandos tales como “MIH Handover Initiated” o “MIH Handover Prepare”.

- Comandos de enlace: originados en MIHF, en nombre de los usuarios MIH, para configurar y controlar las capas inferiores. Son locales y deben ser implementados para interactuar con las tecnologías específicas de acceso.

Servicio de Información Independiente del Medio

El Servicio de Información Independiente del Medio (MIIS) tiene como objetivo el conocimiento de todas las redes heterogéneas que se encuentren disponibles en el área de interés para facilitar el handover a la red idónea en cada momento.

La información que se considera relevante es aquella que ayuda al algoritmo de selección de red a tomar la decisión sobre el handover vertical. Esta información describe diferentes aspectos relacionados con las capas inferiores, tales como mapas de las redes vecinas, zonas de cobertura y otros parámetros del enlace. También se provee información referente a las capas superiores, tales como la disponibilidad de conectividad a Internet o a otro tipo de servicios.

MIIS está diseñado para proveer información sobre redes 802, 3GPP y 3GPP2, aunque esta lista será extendida en el futuro. La información procede de todas las tecnologías disponibles en la zona de interés, no tan sólo de la tecnología a la que se esté conectado. Esto supondrá un ahorro de potencia, ya que no es necesario activar la interfaz correspondiente para obtener información sobre una red específica.

Los Elementos de Información (IE) que provee MIIS se dividen en las siguientes categorías:

- **Información general**: tipo de red, identificador del operador o identificador del proveedor del servicio, entre otros.

- **Información específica de la red de acceso**: características de seguridad, información QoS, costes, etc.

- **Información específica del Punto de Conexión** (PoA, Point of Attachment): dirección MAC, localización geográfica, velocidad de transmisión, rango de los canales, …

- **Información/Servicios de las capas superiores para cada PoA**: número de subredes que soporta el PoA, los métodos de configuración IP disponibles, lista con los servicios soportados, etc.

- **Otra información específica de cada vendedor o servicio**.

Cada MN debe ser capaz de descubrir si una red soporta el nuevo estándar IEEE 802.21. Por otro lado, debe ser capaz de obtener información del MIIS antes de realizar la autentificación para

comprobar los protocolos de seguridad disponibles, la información sobre QoS, etc.

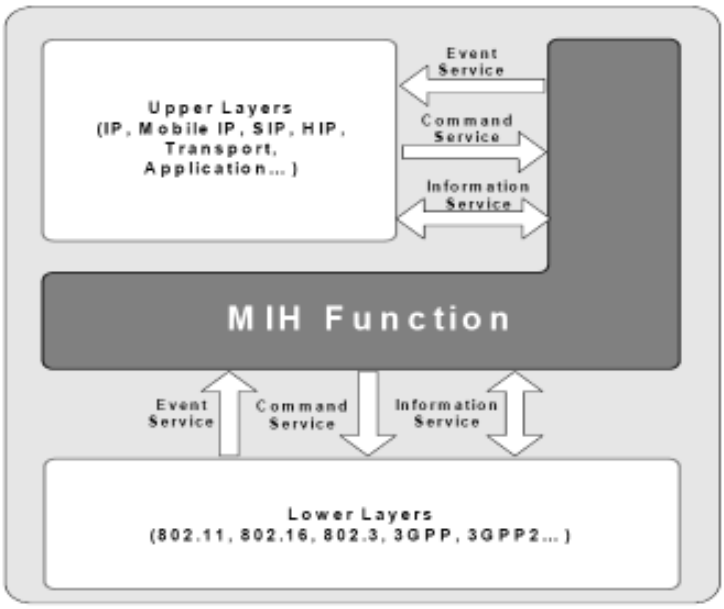

Figura 4.2: Representación de los servicios MIH

4.4 Handover vertical

En este apartado se presenta el procedimiento llevado a cabo para realizar un traspaso desde una red 3G a una red WLAN. A continuación, se detalla el intercambio de mensajes necesario para que el handover sea correcto.

El proceso de handover es iniciado por el usuario MIH en el MN, enviando un mensaje al MIHF local para conocer las redes vecinas disponibles (mensaje 1). El MIHF local solicita esta información al servidor localizado en la red del operador (o en una tercera red, como en este caso en particular) que responde con la información solicitada. A partir de los 4 primeros mensajes, el MN ha sido capaz de obtener la información necesaria sobre las redes vecinas. Como hay disponible una red WLAN, el MN activa la interfaz correspondiente y comienza a escuchar para obtener tramas beacon.

Cuando se recibe la primera trama beacon, la interfaz IEEE 802.11 informa de la disponibilidad del nuevo enlace a través de una

primitiva específica de la tecnología (mensaje 5) que será mapeada en un evento por MIH_LINK_SAP, siendo enviado este evento al usuario MIH en el mensaje 6.

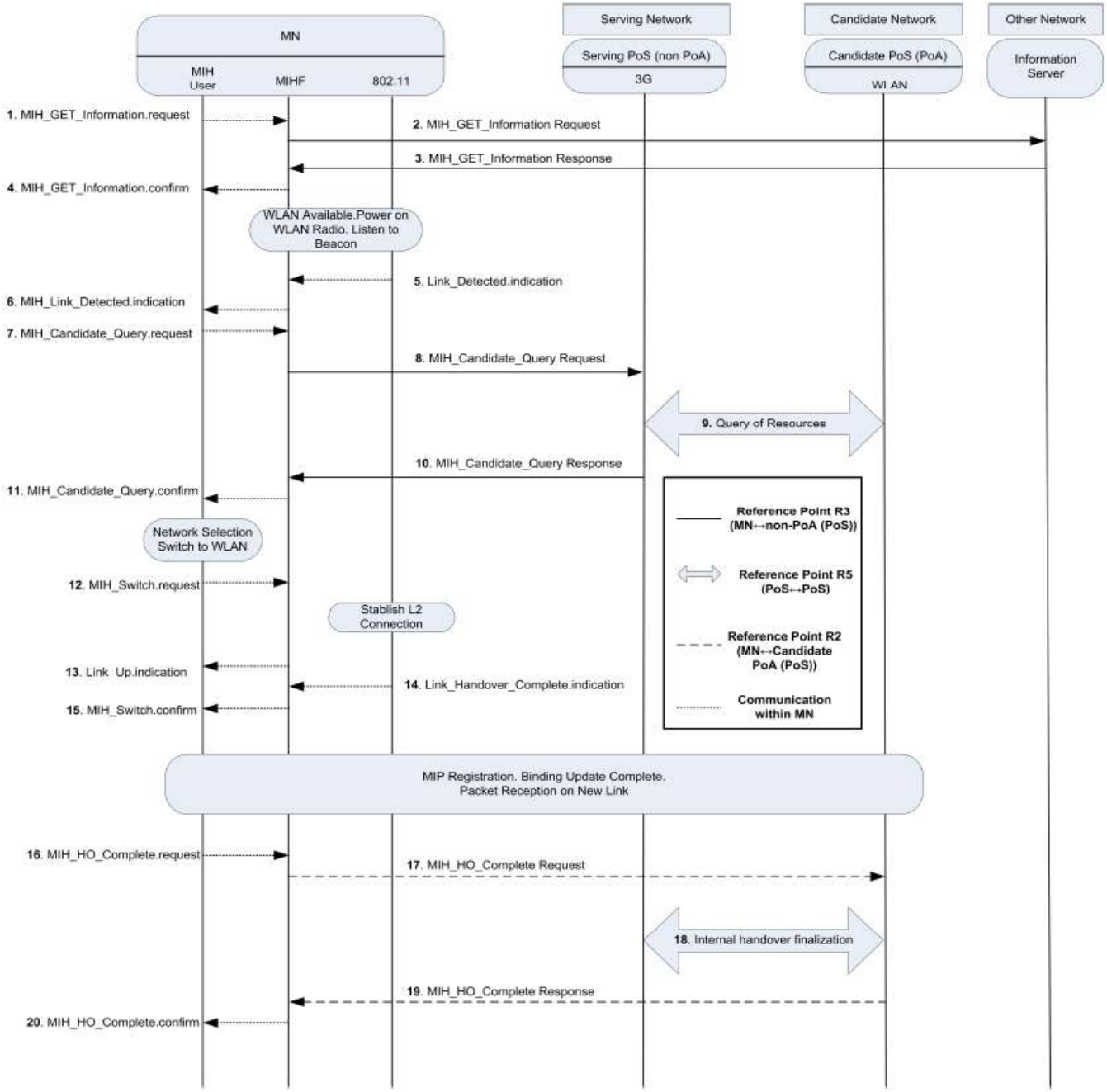

Figura 4.3: Ejemplo de handover entre redes heterogéneas

Cuando el usuario MIH recibe el mensaje de detección de un nuevo enlace, activa el mecanismo de handover enviando al Point of Service (localizado en la red 3G) la información sobre las nuevas redes candidatas. Esto se realiza a través de los mensajes 7 y 8.

Cuando PoS (Point of Service) recibe la información, pregunta a la red candidata la lista de recursos disponibles e informa sobre los requerimientos de QoS del usuario a través del mensaje 9. La respuesta es remitida al usuario MIH a través de los mensajes 10 y 11. En este punto, el MN dispone de la información necesaria como para llevar a cabo la decisión sobre el handover.

Una vez que el usuario MIH decide conectarse a la nueva red, envía un comando a MIHF (mensaje 12) para que realice la conexión a nivel de enlace. MIHF envía un evento al usuario MIH confirmándole la petición de conexión (mensaje 13). Cuando la conexión ha sido establecida, la capa MAC de WLAN envía un mensaje confirmando el fin del traspaso a nivel 2 a través de los mensajes 14 y 15.

Una vez que el mensaje 15 es recibido, se puede comenzar un procedimiento de handover en las capas superiores. En este caso, IP móvil ha sido el utilizado, aunque cualquier otro protocolo de gestión de la movilidad podría haber sido establecido.

Cuando se completa el handover en las capas superiores, el usuario MIH informa a la capa MIHF, encargada de reenviar la información al PoS objetivo (mensajes 16 y 17). En este punto, el nuevo PoS informa a todas las entidades de la red implicadas de que el handover ha sido finalizado (mensaje 18).

Finalmente, el mensaje 19 y 20 cierra el proceso de handover indicando al MN de que todo ha sido correcto.

Para información más detallada sobre este estándar consultar la referencia [1]. Para estudiar el borrador del estándar se remite al lector a la referencia [2].

La implementación práctica de este estándar se está llevando a cabo en un proyecto europeo denominado Daidalos, que actualmente se encuentra en su segunda fase. Este proyecto persigue el desarrollo de un sistema que ofrezca movilidad y QoS independientemente del medio. En el siguiente capítulo se detalla su arquitectura.

Capítulo 5

Daidalos II

DAIDALOS II (**D**esigning **A**dvanced network **I**nterfaces for the **D**elivery and **A**dministration of **L**ocation independent, **O**ptimised personal **S**ervices) es la segunda fase del proyecto europeo Daidalos. Este proyecto surge para rediseñar las comunicaciones, teniendo como objetivo la facilitación del acceso a la información mediante un conjunto de redes heterogéneas que soporten las preferencias del usuario.

Daidalos parte del hecho de que la movilidad se haya convertido en un aspecto fundamental en el mundo de los negocios, la educación y el tiempo libre. Como respuesta al rápido cambio tecnológico sufrido en los últimos años, que ha producido una cierta sensación de complejidad entre los usuarios en el escenario de las comunicaciones, plantea una nueva infraestructura más manejable y útil, en la que desaparecen las fronteras entre redes de diferentes tipos para el usuario gracias a la implementación del handover vertical basado en el estándar IEEE 802.21.

Los conceptos clave que guían a este proyecto son cinco:

- MARQS: Gestión de la movilidad, AAA (Autentificación, Autorización y Cuentas de usuario), Gestión de los recursos, QoS y seguridad.

- VID (Virtual Identity), que separa al usuario del Terminal, ofreciendo privacidad y personalización.

- USP (Ubiquitous and Seamless Pervasiveness), permite adaptarse a cambios en el contexto, movimientos y requerimientos del usuario.

- SIB (Seamless Integration of Broadcast), que integra broadcast tanto a nivel de tecnología como a nivel de servicio.
- Federation, que permite a los operadores de red y proveedores de servicio ofrecer y recibir servicios, dando a los participantes la oportunidad de unirse o dejar cada servicio ofrecido de forma dinámica.

A continuación se detalla las entidades que forman la arquitectura general de Daidalos II.

5.1 Arquitectura general

En esta sección se presenta la arquitectura general de un terminal móvil en Daidalos. Se procederá a la explicación de cada uno de los módulos que componen la arquitectura, así como la comunicación entre las diferentes entidades.

Interface Abstraction Layer (IAL)

IAL modela la Función de Handover Independiente del Medio (MIHF) propuesta por el estándar IEEE 802.21. Su principal objetivo consiste en la comunicación entre las capas superiores e inferiores. Para ello utiliza los tres servicios descritos en el capítulo anterior: el Servicio de Información, el Servicio de Eventos y el Servicio de Comandos.

La comunicación con las capas inferiores se realiza a través de la interfaz MIH_LINK_SAP y con las capas superiores a través de la interfaz MIH_SAP. IAL también puede comunicarse con otras entidades de la red, tanto en la capa 2 (en el caso de APs y estaciones base) como en la capa 3 (routers y otros PoS).

Radio Access Layer (RAL)

RAL define la interfaz entre el módulo IAL y el driver. Hay un RAL para cada tecnología específica, encargándose de gestionar las operaciones radio y siendo responsable de responder a las

peticiones de la capa IAL. También está preparado para informar sobre diferentes aspectos tales como pérdida de la señal, pérdida de la conexión, etc.

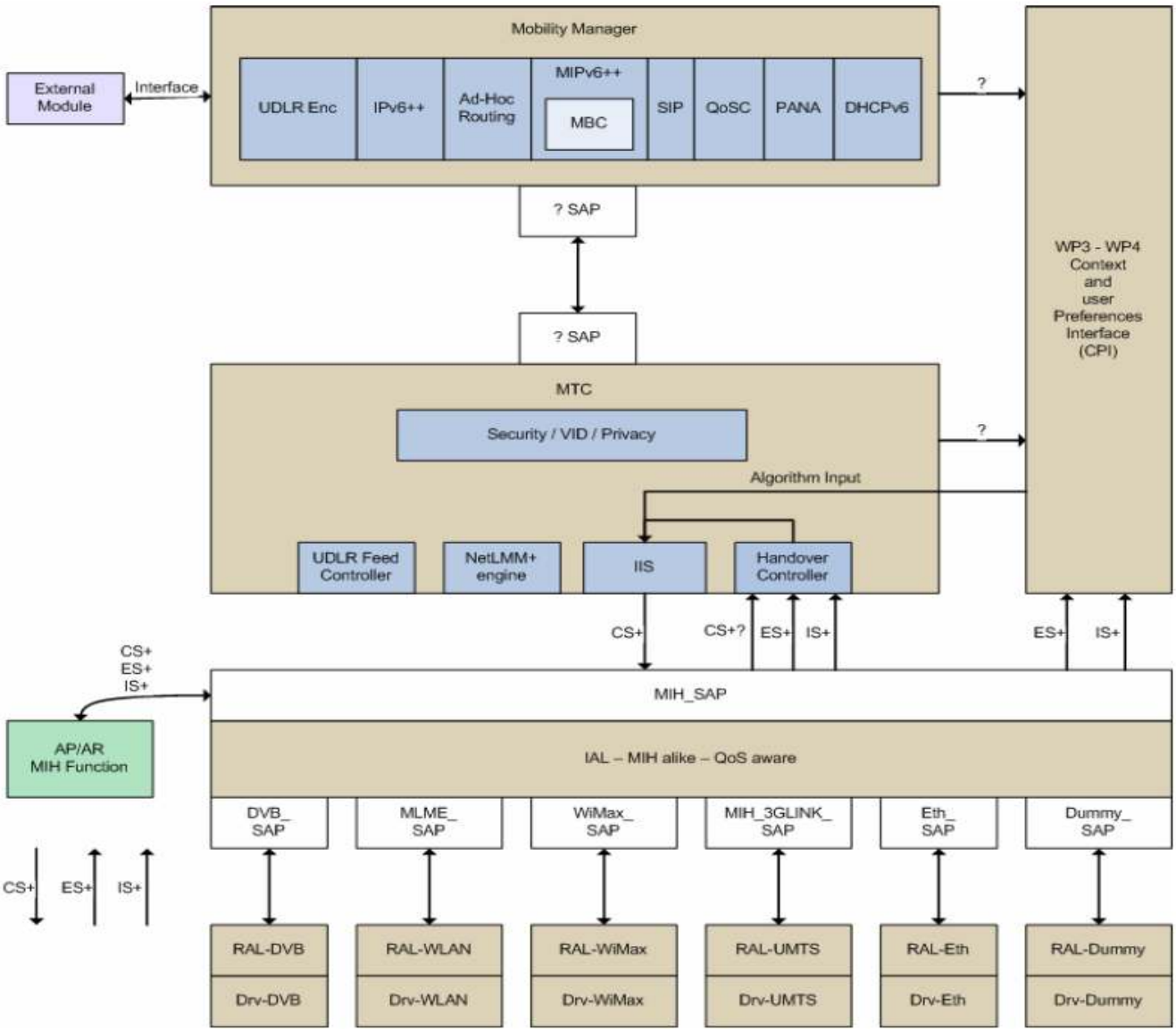

Figura 5.1: Arquitectura general de un nodo móvil

Drivers

Para cada tecnología de acceso hay un módulo del driver responsable de la comunicación con el hardware.

Mobile Terminal Controller (MTC)

Este módulo está compuesto por un Selector Inteligente de la Interfaz, un Controlador de Handover y un módulo que gestiona las funciones de seguridad, VID y privacidad.

El IIS (Intelligent Interface Selector) es la entidad que toma la decisión sobre el control de la interfaz (activación, desactivación) y el handover. Implementa un algoritmo de selección de red basado en los datos recibidos desde dos módulos, el Controlador de Handover y el módulo de Preferencias y Contexto. Una vez que la selección ha sido realizada, comienza el proceso necesario para llevar a cabo el handover.

El Controlador de Handover tiene como principal objetivo el obtener información volátil sobre la red (p. ej. la relación señal a ruido) para ofrecerla al módulo IIS, encargado de la selección de la interfaz.

El módulo de Seguridad/VID/Privacidad es el encargado de la gestión de la seguridad, la privacidad y los VIDs en el módulo MTC, tal y como su nombre indica.

Gestor de la movilidad

Este módulo es el encargado de informar a las aplicaciones sobre los aspectos de la movilidad. Con intención de liberar de peso al módulo MTC, cuando se produce algún cambio de movilidad, esta entidad comunica este cambio al Gestor de movilidad que será el responsable de informar a los módulos implicados.

Esta entidad está formada por una serie de módulos:

- El módulo IPv6, optimiza el tiempo empleado en ciertas operaciones tales como la configuración del enrutamiento y de la dirección.

- El módulo de enrutamiento ad-hoc, representa la interfaz de enrutamiento ad-hoc que será informada de los cambios de movilidad.

- El módulo MIPv6, representa la interfaz MobileIPv6 que será informada de los cambios de movilidad.

- El módulo SIP, representa la interfaz multimedia que será informada de los cambios de movilidad.

- El cliente de Calidad de Servicio (QoSC), representa la interfaz de QoS que será informada de los cambios de movilidad.

- El módulo PANA, representa la interfaz de seguridad que será informada de los cambios de movilidad.

- El codificador UDLR, añade la cabecera GRE al paquete y lo envía al destino correspondiente.

- DHCPv6, esta funcionalidad es requerida para permitir a un MN conectarse a una NEMO (NEtwork that MOves), utilizando una IPv6 que pertenezca a la red que visita.

Context and Preferences Interface (CPI)

Este módulo es la interfaz entre la gestión del contexto y de las preferencias y el resto de módulos de la red.

Se puede encontrar información más detallada sobre la arquitectura general en las referencias [3] y [4].

5.2 Arquitectura QoS

Para el presente proyecto, toma especial importancia la parte de aprovisionamiento de QoS. La arquitectura de QoS es imprescindible no sólo para proveer servicios diferenciados, sino también para el desarrollo de técnicas de mejora y optimización que permitan un uso eficiente de los recursos disponibles en cada momento. De esta forma, el objetivo se convierte en la maximización del número de usuarios y la optimización en el uso de los recursos, sin degradar el servicio, manteniendo el flujo de datos necesario para cada tipo de tráfico.

5.2.1 Conceptos de la arquitectura

En este apartado se detallan los principales conceptos en los que se basa el diseño de la arquitectura.

5.2.1.1 Clasificación a nivel de enlace

Cada RAL (Radio Access Layer) que quiera implementar las funcionalidades de QoS debe ser capaz de clasificar los paquetes, como paso previo para poder gestionar los recursos y asegurar que el QoS negociado no va a ser degradado. La forma en la que los paquetes son clasificados varía entre las tecnologías IEEE 802 y las que no lo son.

En las tecnologías que no pertenecen al IEEE 802, tales como UMTS o DVB, el tráfico se clasifica usando la dirección IPv6 del terminal móvil y el campo DSCP (Differentiated Services Code Point).

En las redes IEEE 802, hay un gran número de terminales que tan sólo implementan hasta el nivel 2, tales como APs y switches Ethernet. Debido a esto, no es seguro que estos terminales sean capaces de clasificar el tráfico de acuerdo a los campos de la cabecera de IPv6. La solución adoptada consiste en el uso de 802.1q, con un campo de prioridad del usuario marcado de acuerdo al DSCP de IPv6. De esta forma, el tráfico será clasificado en función de tres valores: dirección MAC origen, dirección MAC destino y campo de prioridad de usuario en 802.1q.

5.2.1.2 Gestión de los recursos

Esencialmente, cinco tipos de primitivas han sido concebidas para proveer soporte de QoS a nivel 2 en redes heterogéneas:

- Reserva QoS
- Petición de recursos
- Notificación de degradación de QoS
- Movilidad
- Multicast

QoS y Movilidad

Durante el handover, es esencial mantener la QoS en la medida de lo posible, desde que el terminal abandona el AP al que estaba conectado hasta que se consigue la conexión con el nuevo. De hecho, la reserva de recursos para QoS puede llevar hasta 100 milisegundos para algunas tecnologías, lo que puede tener un impacto notable en la QoS percibida, por ejemplo durante una conversación de voz.

Una posible solución para mantener la QoS en los handovers consiste en dividir el proceso en dos pasos: preparación y ejecución. En la preparación se informa a los PoAs involucrados para que reserven los recursos necesarios con antelación, de manera que cuando el terminal móvil consiga la conexión, sólo haya que activar los recursos previamente reservados. En la fase de ejecución, las capas superiores deben informar a la capa QoS Controller en el momento exacto en el que finalice el handover.

Handover iniciado por la red

Daidalos implementa un módulo llamado Performance Management (PM), encargado de gestionar la red. Su funcionalidad básica consiste en solicitar un traspaso si las condiciones de la red son insuficientes para el servicio prestado.

Una mejora importante que implementa el proyecto Daidalos II en este módulo consiste en el soporte del multihoming. Esto permite al algoritmo PM considerar cada servicio, o incluso a los enlaces de subida y bajada de cada servicio que se ejecuten en un terminal móvil, como entidades separadas que pueden ser servidas a través de diferentes canales físicos. Por ejemplo, si hay una llamada de voz y un servicio descargando datos sobre TD-CDMA y la red se sobrecarga, el PM puede mover a WLAN sólo el segundo servicio, ya que las redes TD-CDMA son preferidas para las llamadas de voz por sus características. Como se aprecia a través del ejemplo, este sistema ofrece una gran flexibilidad, proporcionando una gran mejora en el servicio prestado al usuario.

A continuación se describen los módulos e interfaces que componen la arquitectura de QoS en Daidalos.

5.2.2 Diseño de la arquitectura e interfaces

La arquitectura genérica, centrándose en QoS, se muestra en la figura 5.2.

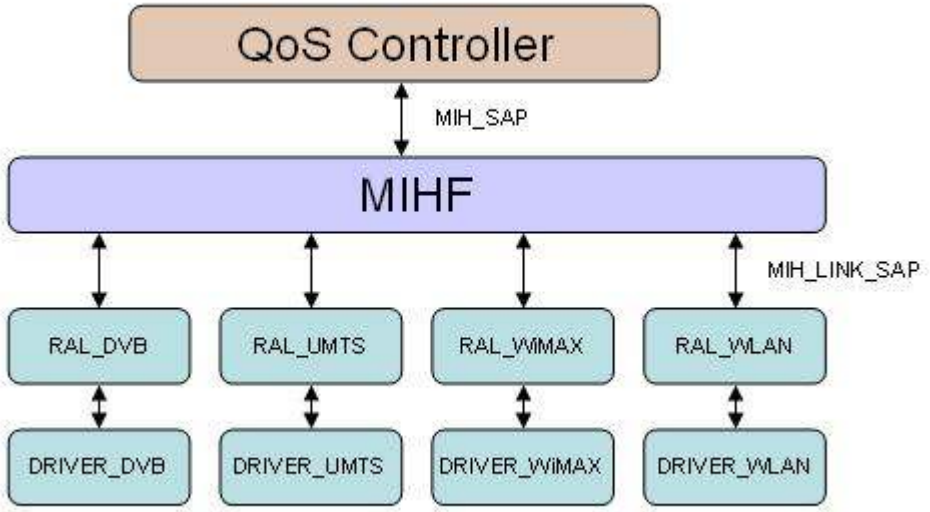

Figura 5.2: Arquitectura QoS Daidalos

A continuación se detallan las entidades más importantes que forman la arquitectura de cada nodo, recorriendo la estructura de arriba hacia abajo.

La capa QoS Controller provee una interfaz de servicio que une el módulo de nivel 3, QoS Manager, con la capa MIHF, incluyendo un manejo transparente de las complejas topologías a nivel 2 y la traducción de las direcciones IPv6.

La capa MIHF ha sido ampliamente comentada en el capítulo anterior correspondiente al estándar IEEE 802.21. Como se dijo, se trata de una capa independiente del medio, que abstrae a las capas superiores de las tecnologías específicas de acceso.

Como se puede apreciar en la figura, la arquitectura incluye un RAL para cada interfaz disponible en cada nodo. El módulo RAL implementa la interfaz MIH_LINK_SAP, que permite la comunicación con MIHF, extendida con algunos comandos y eventos que permiten la implementación de QoS.

El presente proyecto se enmarca en el desarrollo de la capa RAL_WLAN, implementando las funcionalidades que debe de llevar a cabo este módulo e interpretando las diferentes primitivas que comunican a RAL con el resto de componentes de la arquitectura.

La función de QoS de RAL_WLAN se basa en el mecanismo de acceso EDCA, descrito por el estándar 802.11e recientemente aprobado. Consiste en la diferenciación de servicio a partir de un conjunto de parámetros (principalmente, CWmin, CWmax, AIFS y TXOP).

La siguiente figura muestra una versión simplificada de una arquitectura MT-PoA centrándose en la parte de QoS. Las principales funcionalidades del terminal móvil son dos: informar sobre el estado del canal radio al punto de acceso (obteniendo esta información del driver) y configurar el conjunto de parámetros EDCA que recibirá del AP en las interfaces wireless.

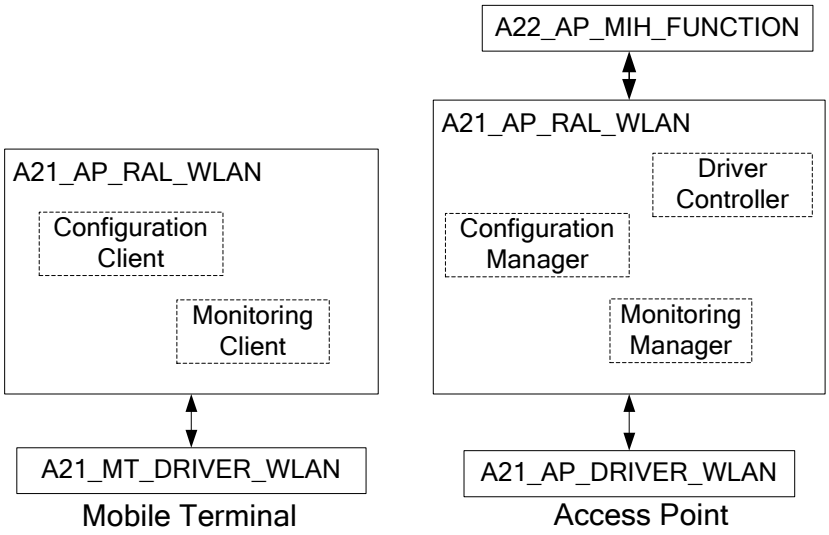

Figura 5.3: Arquitectura QoS en WLAN

Las principales funcionalidades del módulo RAL en el AP son: recolectar información sobre el estado de WLAN y configurar los parámetros EDCA, dependiendo de los requerimientos de QoS. Cuando llega al AP una petición nueva, se utiliza toda la información disponible para establecer el conjunto de parámetros idóneo, distribuyéndolo a todas las estaciones conectadas al punto de acceso, en caso de que sea posible aceptar el nuevo flujo. Si no, el control de admisión decidirá rechazar este flujo entrante.

Por último, el módulo DRIVER_WLAN representa el driver que controla la tarjeta WLAN 802.11e, basado en el driver MADWIFI. Las funcionalidades implementadas por esta entidad son, básicamente, las necesitadas por el módulo RAL_WLAN: por una parte, proveer información referente al estado del canal wireless y por otro lado, proporcionar una interfaz que soporte la re-configuración de la tarjeta para adaptarse a las condiciones de la red.

Aparte de WLAN, Daidalos contempla la posibilidad de utilizar otras tecnologías: DVB, UMTS y WiMAX.

Para una descripción más detallada de la arquitectura y diseño de QoS en Daidalos, se remite al lector a [5].

Una vez que se conoce con detalle el estado del arte de las tecnologías implicadas en el proyecto, se puede proceder a la implementación de la subcapa RAL_WLAN. Como se ha citado anteriormente, este módulo debe implementar la siguiente funcionalidad:

- En el **AP**, al recibir un conjunto de parámetros EDCA de las capas superiores, debe distribuir este conjunto a todas las estaciones registradas en la red y configurar estos parámetros en su máquina.

- En los **NM**, al recibir la primitiva correspondiente, el nodo escuchará el medio para recibir un conjunto de parámetros EDCA del AP y los configurará en su máquina.

En la siguiente figura se muestra un esquema general del sistema a implementar.

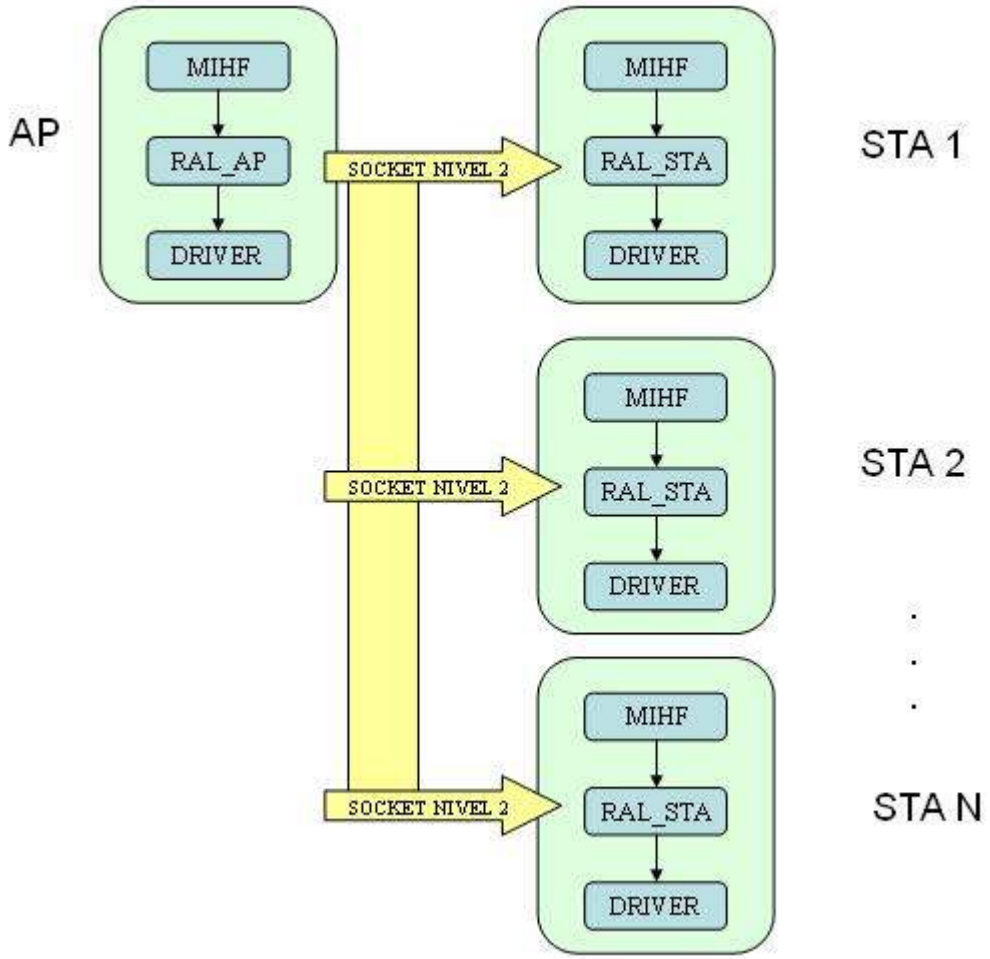

Figura 5.4: Esquema general de la funcionalidad a implementar

Se comenzará por el estudio del driver Madwifi, prestando especial atención a la modificación de los parámetros EDCA. A continuación, se detallará el proceso a seguir para la distribución de los parámetros a través de la WLAN por medio de sockets de nivel 2. Por último, se describirán las primitivas que comunican a los módulos MIHF y RAL.

Parte III

Desarrollo de una Capa de Abstracción para el Nivel Radio

Capítulo 6

Controlador 802.11e EDCA

La implementación del protocolo IEEE 802.11e llevará un tiempo. En la actualidad, pocos controladores funcionan bajo este estándar. Gracias a proyectos anteriores, en los que se ha estudiado en profundidad la respuesta de varios drivers a estas especificaciones, se sabe que el driver Madwifi ofrece un gran soporte del mecanismo EDCA, por lo que será el utilizado. A esto se añade la ventaja de ser un programa de código abierto, por lo que se dispondrá de él para cualquier modificación que sea necesaria.

En este capítulo se describe el proceso seguido para la modificación de los parámetros EDCA. Se comienza por el estudio del driver Madwifi, ahondando en cómo este driver inicializa y permite modificar los parámetros que nos permiten ofrecer QoS. A continuación, se detalla el uso de la función de Linux ioctl(), que permite la modificación de estos parámetros sin necesidad de estar recompilando el driver cada vez que se produzca un cambio.

6.1 Driver Madwifi

Un driver o controlador de dispositivo es un programa que permite al sistema operativo interactuar con un periférico, haciendo una abstracción del hardware y proporcionando una interfaz (posiblemente estandarizada) para usarlo. Se puede esquematizar como un libro de instrucciones que le indica al sistema operativo como debe controlar y comunicarse con un dispositivo en particular. Se trata, por tanto, de una pieza esencial sin la que no se podría usar el hardware.

En el sistema operativo Linux, un driver puede actuar como módulo o estar integrado en el núcleo (kernel) del sistema operativo. Si está integrado en el kernel, el driver estará siempre dispuesto para actuar, ya que es parte del núcleo. Por el contrario, si actúa como módulo, el driver podrá cargarse en tiempo real en el núcleo del sistema operativo sin necesidad de recompilar el núcleo. En palabras más técnicas, un driver como módulo es un código objeto no asociado a un fichero ejecutable que puede enlazarse de manera dinámica con el núcleo cuando este está en funcionamiento. La principal utilidad de los módulos es que se pueden realizar modificaciones sin necesidad de recompilar el núcleo del sistema. Esta opción es muy útil para las características de este proyecto.

Madwifi es la abreviatura de Multiband Atheros Driver for Wifi. Se trata de un proyecto de código libre para dotar al sistema operativo Linux de un controlador para dispositivos inalámbricos basados en el chipset de Atheros. Muestra como una interfaz más la tarjeta WLAN y permite su configuración mediante herramientas genéricas como ifconfig y específicas de las redes inalámbricas como iwconfig, iwlist, iwpriv, etc.

Madwifi es un driver que opera como módulo (aunque se plantea en la actualidad la posibilidad de integrarlo en el kernel), pudiéndose cargar y eliminar mediante los comandos Linux `modprobe` y `rmmod` respectivamente. Esta característica nos va a facilitar enormemente la modificación del driver por las ventajas citadas anteriormente.

Otra funcionalidad importante es que permite que las tarjetas funcionen como estaciones, AP o como monitores de red. Da soporte para las Wireless Tools y tiene implementado el cifrado WPA/WEP y la autentificación en el AP.

Para cumplir con el objetivo del proyecto es necesaria la modificación del driver. Por ello se estudió en profundidad la estructura del código. A continuación se esquematizan las partes más importantes de Madwifi:

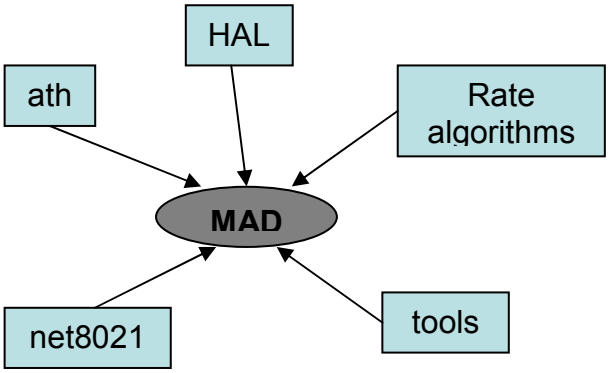

Figura 6.1: Estructura del código de Madwifi

El código consta principalmente de 5 partes:

- **ath**: define las llamadas específicas de Atheros a la capa net80211 y los accesos al hardware a través de HAL.

- **HAL**: Hardware Abstraction Layer. Todos los accesos al hardware deben ir a través de este componente de código cerrado mantenido por Atheros. Desafortunadamente no hay documentación sobre este código, excepto las interfaces públicas definidas en *hal/ah.h*.

- **rate algorithms**: diferentes algoritmos para seleccionar la mejor velocidad de transmisión han sido implementados por los módulos rate en ath_rate.

- **net80211**: contiene funciones genéricas de 802.11.

- **tools**: contiene varias herramientas muy útiles para el manejo del driver, entre las que destacan: athchans, para aplicar una máscara de canales al escaneo; athstats, que aporta información sobre la transmisión (paquetes recibidos, CRC, …) y athctrl , que establece los tiempos ACK y CTS basándose en la distancia entre dos estaciones.

A continuación se explica con mayor detalle la parte HAL ya que se trata de una parte muy importante de este driver.

HAL

HAL son las iniciales de Hardware Abstraction Layer. Es similar al firmware de una tarjeta, con la diferencia de no estar almacenado en ella. HAL es un módulo del kernel que implementa una API (Application Programming Interface), es decir, una interfaz entre el hardware y el driver. No es exactamente como un firmware, porque 'firmware' es un término que generalmente se refiere al código ejecutado por DSPs o procesadores especiales dentro de la placa.

HAL es código cerrado y sólo está disponible el binario compilado para las diferentes arquitecturas de los ordenadores. Esto es debido a que los chipsets de Atheros pueden sintonizar un amplio rango de frecuencias, parte de ellas fuera de la banda permitida para uso sin licencia (banda ISM). Las frecuencias no ISM pertenecen a varias agencias con diferentes propósitos, tales como el uso militar o el radar para uso civil. Además, existen limitaciones en la potencia de transmisión permitida. Para asegurar que se cumplen todas estas limitaciones, Madwifi las implementa en esta parte de código cerrado, asegurando que los registros hardware y los accesos a memoria se realizan en la parte de código abierto para poder comprobar que no actúa de manera sospechosa.

La principal funcionalidad que se requiere a la aplicación que tiene como objetivo este proyecto es la configuración de los parámetros EDCA. A continuación se describe detalladamente dónde inicializa el driver estos parámetros y cómo es posible modificarlos de forma correcta a través del método que nos proporciona HAL.

6.2 Parámetros EDCA

Los parámetros EDCA están definidos en Madwifi como una estructura. Las estructuras en C agrupan a una o más variables, generalmente de diferentes tipos, bajo un mismo nombre para hacer

más fácil su manejo. Dicha estructura se encuentra definida en la cabecera *net80211/ieee80211_proto.h* como:

```
struct wmeParams {
   u_int8_t wmep_acm;              /* ACM parameter */
   u_int8_t wmep_aifsn;            /* AIFSN parameters */
   u_int8_t wmep_logcwmin;         /* cwmin in exponential form */
   u_int8_t wmep_logcwmax;         /* cwmax in exponential form */
   u_int16_t wmep_txopLimit;       /* txopLimit */
   u_int8_t wmep_noackPolicy;      /* No-Ack Policy: 0=ack, 1=no-ack */
}
```

Esta estructura es parte de estructuras más genéricas, quedando enmarcada en el siguiente lugar:

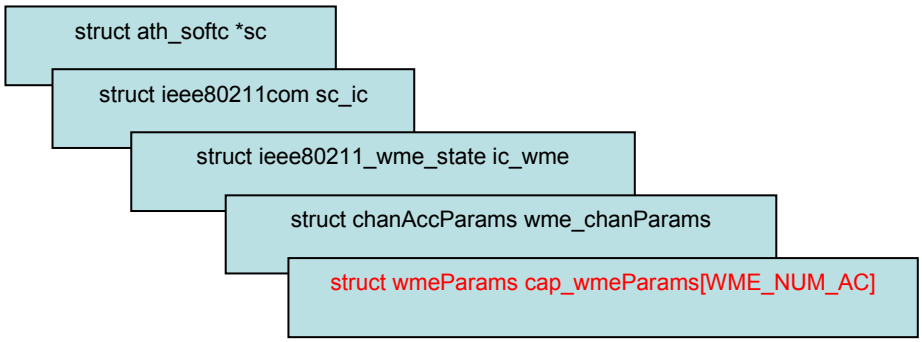

Figura 6.2: Jerarquía de la estructura wmeParams

Inicialización de los parámetros EDCA

La inicialización de los parámetros se lleva a cabo en el siguiente método en el archivo *net80211/ieee80211_proto.c*:

```
void ieee80211_wme_initparams(struct ieee80211vap *vap)
{
   struct ieee80211com *ic = vap->iv_ic;

   IEEE80211_LOCK(ic);
   ieee80211_wme_initparams_locked(vap);
   IEEE80211_UNLOCK(ic);
}
```

Como se puede apreciar, se cierra el acceso a los parámetros mientras están siendo modificados a través de `IEEE80211_LOCK(ic)` para que nadie pueda acceder a ellos al mismo tiempo y así evitar problemas.

En el método llamado, `ieee80211_wme_initparams_locked(vap)`, se encuentra la siguiente estructura donde se definen los valores por defecto para los diferentes parámetros EDCA. Por ejemplo, para el tráfico de voz:

```
    static struct phyParamType bssPhyParamForAC_VO[IEEE80211_MODE_MAX]
= {
          /* IEEE80211_MODE_AUTO  */ { 2, 2,  3,  47, 0 },
          /* IEEE80211_MODE_11A   */ { 2, 2,  3,  47, 0 },
          /* IEEE80211_MODE_11B   */ { 2, 3,  4, 102, 0 },
          /* IEEE80211_MODE_11G   */ { 2, 2,  3,  47, 0 },
          /* IEEE80211_MODE_FH    */ { 2, 3,  4, 102, 0 },
          /* IEEE80211_MODE_TURBO */ { 1, 2,  2,  47, 0 },
          /* IEEE80211_MODE_TURBO */ { 1, 2,  2,  47, 0 }};
```

El siguiente cuadro recoge los valores por defecto de CWmin, CWmax, AIFS y TXOP para los distintos tipos de tráfico en el modo de funcionamiento IEEE 802.11b.

	AC_VO	AC_VI	AC_BE	AC_BK
Cwmin	3	4	5	5
Cwmax	4	5	10	10
AIFS	2	2	7	2
TXOP	3264	6016	0	2048

Cuadro 6.1: Parámetros por defecto en el modo 802.11b

Se pueden modificar los valores con los que los parámetros son inicializados tan sólo con modificar los números en la estructura anterior.

Modificación de los parámetros EDCA

Por otra parte, la modificación de los parámetros desde el código se puede realizar a través del método definido en *ath/if_ath.c*

```
    static int ath_wme_update(struct ieee80211com *ic)
```

, asignando los nuevos valores en la estructura `struct wmeParams`, subestructura de `struct ieee80211com *ic` (como se ha citado anteriormente), que será pasada como argumento al método de actualización de los parámetros.

Si se desea modificar los parámetros de acceso al medio, estos deben ser modificados en la tarjeta, ya que es la encargada del control del medio físico y por tanto estos valores se encuentran en registros internos de la tarjeta. Para ello, el método `ath_wme_update` utiliza la siguiente función proporcionada por HAL (*hal/ah.h*), interfaz entre el hardware y el driver:

```
    HAL_BOOL   __ahdecl(*ah_setTxQueueProps)(struct ath_hal *, int q,
const HAL_TXQ_INFO *qInfo);
```

El método traduce de la siguiente forma en *ath/if_athvar.h*:

```
#define  ath_hal_settxqueueprops(_ah, _q, _qi) \
((*(_ah)->ah_setTxQueueProps)((_ah), (_q), (_qi)))
```

, definiéndose el método utilizado finalmente como:

```
ath_hal_settxqueueprops(ah, txq->axq_qnum, &qi)
```

Los argumentos que se le pasan tienen el siguiente significado:

- ah es la instancia del módulo HAL que se está utilizando.
- txq->axq_qnum indica cual de las 4 ACs es modificada.
- &qi apunta a la estructura de tipo HAL_TXQ_INFO que define los nuevos valores a establecer.

A continuación, se adjunta la estructura HAL_TXQ_INFO definida en *hal/ah.h* para observar que parámetros contiene:

```
typedef struct {
    u_int32_t    tqi_ver;           /* hal TXQ version */
    HAL_TX_QUEUE_SUBTYPE tqi_subtype;    /* subtype if applicable */
    HAL_TX_QUEUE_FLAGS tqi_qflags;       /* flags (see above) */
    u_int32_t    tqi_priority;      /* (not used) */
    u_int32_t    tqi_aifs;          /* aifs */
    u_int32_t    tqi_cwmin;         /* cwMin */
    u_int32_t    tqi_cwmax;         /* cwMax */
    u_int16_t    tqi_shretry;       /* rts retry limit */
    u_int16_t    tqi_lgretry;       /* long retry limit (not used)*/
    u_int32_t    tqi_cbrPeriod;     /* CBR period (us) */
    u_int32_t    tqi_cbrOverflowLimit;    /* threshold for CBROVF int */
    u_int32_t    tqi_burstTime;     /* max burst duration (us) */
    u_int32_t    tqi_readyTime;     /* frame schedule time (us) */
    u_int32_t    tqi_compBuf;       /* comp buffer phys addr */
} HAL_TXQ_INFO;
```

Mediante el uso del método `ath_wme_update(struct ieee80211com *ic)` se pueden modificar los parámetros en el código. Compilando el driver, descargando y cargando los módulos se utilizarían los nuevos valores. Sin embargo, este proceso es bastante lento, por lo que interesaría encontrar la forma de modificar los parámetros "en caliente", es decir, desde la línea de comandos sin necesidad de generar un nuevo módulo. La respuesta a esta necesidad la da la función de Linux ioctl (). A continuación se detalla el proceso seguido para el uso de esta función.

6.3 Modificación a través de ioctl

La función ioctl abre una puerta entre el módulo cargado en el kernel y el espacio de usuario permitiendo la comunicación con el driver sin necesidad de recompilar el código. Esta característica es muy útil para la aplicación a implementar, ya que facilitará y agilizará el proceso de modificación de los parámetros EDCA.

El funcionamiento del método ioctl () se ilustra en la siguiente figura.

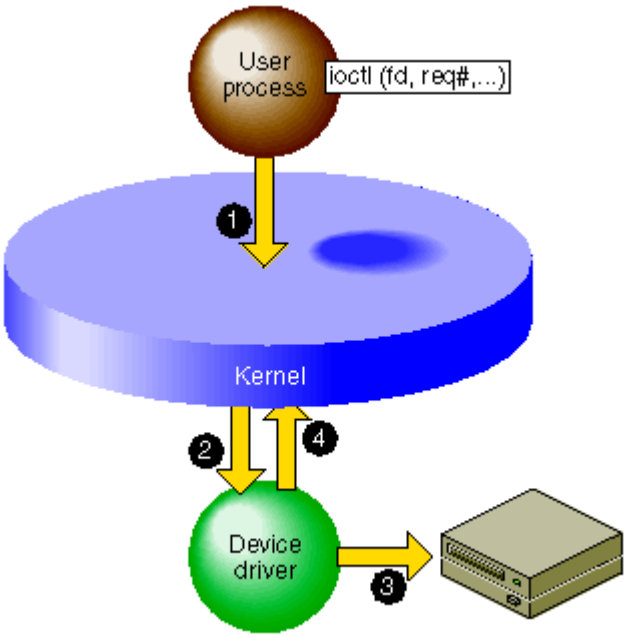

Figura 6.3: Funcionamiento de la función ioctl ()

El funcionamiento básico es el siguiente:

1.- El proceso de usuario llama a la función del kernel ioctl (), pasándole los argumentos necesarios.

2.- El kernel se comunica con el driver para la funcionalidad que sea requerida.

3.- El driver interpreta los parámetros y envía los comandos necesarios al dispositivo.

4.- El driver devuelve al kernel un código de salida con el resultado de la operación realizada.

Para más información sobre el funcionamiento de ioctl consultar la referencia [6].

Implementación

El primer paso para la creación de la ioctl consiste en definir una estructura que será utilizada para pasar los parámetros al método. Será definida en *ath/if_athioctl.h* y contendrá el valor de CWmin, CWmax, AIFS y TXOP. También deberá declarar sobre que tipo de cola se quiere realizar la modificación, al existir cuatro tipos diferentes.

```
struct ath_queue {
   char ad_name[IFNAMSIZ]; // Interface name, e.g "ath0"
   u_int8_t ac;
   u_int8_t haltype;
   int32_t cwmin;
   int32_t cwmax;
   int8_t aifs;
   int32_t txop;
};
```

Además, hay que definir de qué forma será llamada la ioctl. Para ello es necesario declarar el nombre en *ath/if_athioctl.h*:

```
#define SIOCGATHQUEUE      (SIOCDEVPRIVATE+3)
#define  SIOCGATHQUEUE     _IOWR('i', 139, struct ath_queue)
```

El siguiente paso consiste en añadir el nuevo caso al método que captura las ioctl (ath_ioctl) en el fichero *ath/if_ath.c* para que cuando reciba la ioctl creada llame al nuevo método ath_ioctl_queue, que se creará a continuación.

```
case SIOCGATHQUEUE:
      error = ath_ioctl_queue(sc, (struct ath_queue *) ifr);
      break;
```

El nuevo método se declara como sigue:

```
int ath_ioctl_queue(struct ath_softc *sc, struct ath_queue *queue)
```

, donde se asignan los nuevos valores definidos en la estructura `struct ath_queue *queue` a la estructura `struct wmeParams`. A continuación, se llama al método `ath_wme_update(ic)` encargado de actualizar los parámetros a través del método que ofrece HAL para interactuar con el hardware.

De esta forma hemos conseguido establecer el nuevo conjunto de parámetros sin necesidad de realizar la tediosa tarea de compilar y cargar los módulos.

Capítulo 7

Distribución de parámetros en una misma WLAN

Una de las funciones más importantes que tiene que implementar la capa RAL es la distribución de los parámetros EDCA. El AP recibe la configuración y se encarga de distribuirla a todas las estaciones conectadas a él. El envío se realiza a través de un mecanismo llamado socket que se explica con más detalle a continuación.

7.1 Socket

Un socket es una interfaz de entrada y salida de datos que permite la intercomunicación entre procesos. Los procesos pueden estar ejecutándose en el mismo o en distintos sistemas, conectados mediante una red.

Para crear un socket basta con llamar a la siguiente función:

```
int socket ( int domain, int type, int protocol )
```

A continuación se detalla el significado de cada argumento:

int domain

Los sockets se crean dentro de un dominio de comunicación, que nos dice donde se encuentran los procesos que se van a comunicar. Si los procesos se encuentran en el mismo sistema, el dominio de comunicación será PF_UNIX, si los procesos están en sistemas diferentes y se hayan conectados a través de una red

TCP/IP el dominio de comunicación será PF_INET. En nuestro caso, al trabajar en el nivel 2, será necesario utilizar un dominio de comunicación diferente, PF_PACKET.

PF_PACKET permite a una aplicación enviar y recibir paquetes directamente en la interfaz, evitando el procesamiento de cabeceras (como por ejemplo en TCP o UDP), es decir, cualquier paquete enviado por este socket pasará directamente a la interfaz correspondiente y cualquier paquete recibido en el socket será pasado directamente a la aplicación.

int type

El tipo de socket viene predeterminado por el dominio de comunicación elegido. PF_PACKET soporta dos tipos: SOCK_DGRAM y SOCK_RAW.

SOCK_DGRAM para paquetes con la cabecera de la capa de enlace eliminada.

SOCK_RAW para paquetes completos de la capa de enlace.

En nuestra aplicación utilizaremos el tipo SOCK_DGRAM.

int protocol

El protocolo debe especificarse dentro de los definidos en el fichero */usr/include/linux/if_ether.h* que representa los protocolos registrados que pueden ser utilizados en una trama Ethernet.

Debido al gran potencial de la familia PF_PACKET sólo es posible crear sockets de este dominio teniendo permisos de root.

Una vez creado el socket hay que configurarlo para la aplicación específica. Para ello se utiliza la estructura sockaddr_ll definida como:

```
struct sockaddr_ll {
        unsigned short  sll_family;     /* Always AF_PACKET */
        unsigned short  sll_protocol;  /* Physical layer protocol */
        int             sll_ifindex;    /* Interface number */
        unsigned short  sll_hatype;     /* Header type */
        unsigned char   sll_pkttype;    /* Packet type */
        unsigned char   sll_halen;      /* Length of address */
        unsigned char   sll_addr[8];    /* Physical layer address */
};
```

En el campo sll_addr se escribirá la dirección MAC de la máquina destino.

Para establecer el número de la interfaz se utiliza la ioctl SIOCGIFINDEX, que devuelve el índice de la interfaz dado su nombre. Este proceso es necesario ya que el número varía al ser asignado por el sistema operativo cada vez que la interfaz es reiniciada.

El siguiente paso consiste en llamar a la función sendto() para enviar los datos deseados, en nuestro caso la estructura que contiene los parámetros EDCA:

```
    sent = sendto(sock, &wme_param,sizeof(struct ath_queue) , 0,
(struct sockaddr *) &dest, sizeof(dest));
```

El primer parámetro especifica un descriptor de socket, el segundo es un puntero a los datos a enviar, el tercero designa el tamaño de estos datos, el cuarto permite el uso de flags (en la aplicación presente no han sido utilizados), el quinto especifica la estructura que contiene la dirección destino y el sexto su tamaño.

En el sistema que recibe los datos se crea el mismo socket que en el caso anterior y se llama a la función recvfrom():

```
    received = recvfrom(sock, &wme_param, sizeof(struct ath_queue), 0,
(struct sockaddr *) &from, &fromlen);
```

En este caso el primer parámetro especifica el descriptor de socket (del mismo tipo que el creado en el servidor), el segundo es un puntero a un buffer donde guardar los datos recibidos, el tercero designa el tamaño de ese buffer, el cuarto permite el uso de flags, el quinto especifica una estructura donde guardar la dirección del emisor y el sexto su tamaño.

Una vez que se dispone de las herramientas necesarias para la modificación y distribución de los parámetros, se procede a la

comunicación entre los módulos MIHF y RAL a través de las primitivas que el proyecto Daidalos define para esta función.

Capítulo 8

Interfaz MIHF - RAL

Los distintos módulos que componen la pila de protocolos de Daidalos deben ser capaces de comunicarse entre sí. Esta comunicación se realiza a través de una serie de primitivas bien definidas.

Para finalizar la implementación del módulo RAL_WLAN, es necesario que esta entidad sea capaz de reconocer las primitivas procedentes de MIHF y actué en consecuencia, enviando confirmaciones cuando sea necesario.

En este capítulo se detallan los pasos seguidos para implementar las funcionalidades requeridas por las primitivas de la parte de QoS procedentes del módulo MIHF. Se comenzará por detallar cuales son estas primitivas y cual es su significado, procediendo luego a la implementación de las mismas en el código del módulo RAL_WLAN.

8.1 Primitivas MIHF – RAL

Como se ha citado anteriormente, las primitivas son métodos bien definidos que sirven para la comunicación entre módulos diferentes. A continuación se detallan las primitivas declaradas para proporcionar QoS en redes heterogéneas.

Origen	Destino	Primitiva	Parámetros
A22_AR_MIH_Function	A21_AR_RAL_WLAN	Link_Resource_Prepare.Request	– IP@ – QoS param
A21_AR_RAL_WLAN	A22_AR_MIH_Function	Link_Resource_Prepare.Confirmation	– Cnx_Id
A22_AR_MIH_Function	A21_AR_RAL_WLAN	Link_Resource_Activate.Request	– Cnx_Id
A21_AR_RAL_WLAN	A22_AR_MIH_Function	Link_Resource_Activate.Confirmation	– Cnx_Id
A22_AR_MIH_Function	A21_AR_RAL_WLAN	Link_Resource_Deactivate.Request	– Cnx_Id
A21_AR_RAL_WLAN	A22_AR_MIH_Function	Link_Resource_Deactivate.Confirmation	– Cnx_Id

Cuadro 8.1: Interfaz A22_AR_MIH_Function::A21_AR_RAL_WLAN

Cada primitiva cumple una funcionalidad específica:

- **Link_Resource_Prepare.Request** (MIHF-> RAL): reserva los recursos necesarios para ofrecer la QoS negociada.
- **Link_Resource_Prepare.Confirmation** (RAL-> MIHF): confirma que la reserva de recursos se ha realizado de forma correcta.
- **Link_Resource_Activate.Request** (MIHF-> RAL): una vez que la reserva de los recursos ha sido utilizada, MIHF puede hacer uso de ellos.
- **Link_Resource_Activate.Confirmation** (RAL-> MIHF): confirmación de la primitiva anterior.
- **Link_Resource_Deactivate.Request** (MIHF-> RAL): libera los recursos previamente reservados para que puedan ser utilizados por una nueva petición.
- **Link_Resource_Deactivate.Confirmation** (RAL-> MIHF): confirma la liberación de los recursos.

En la siguiente figura se muestra el intercambio de primitivas entre los módulos MIHF y RAL en un AP y las funcionalidades que conlleva cada mensaje recibido.

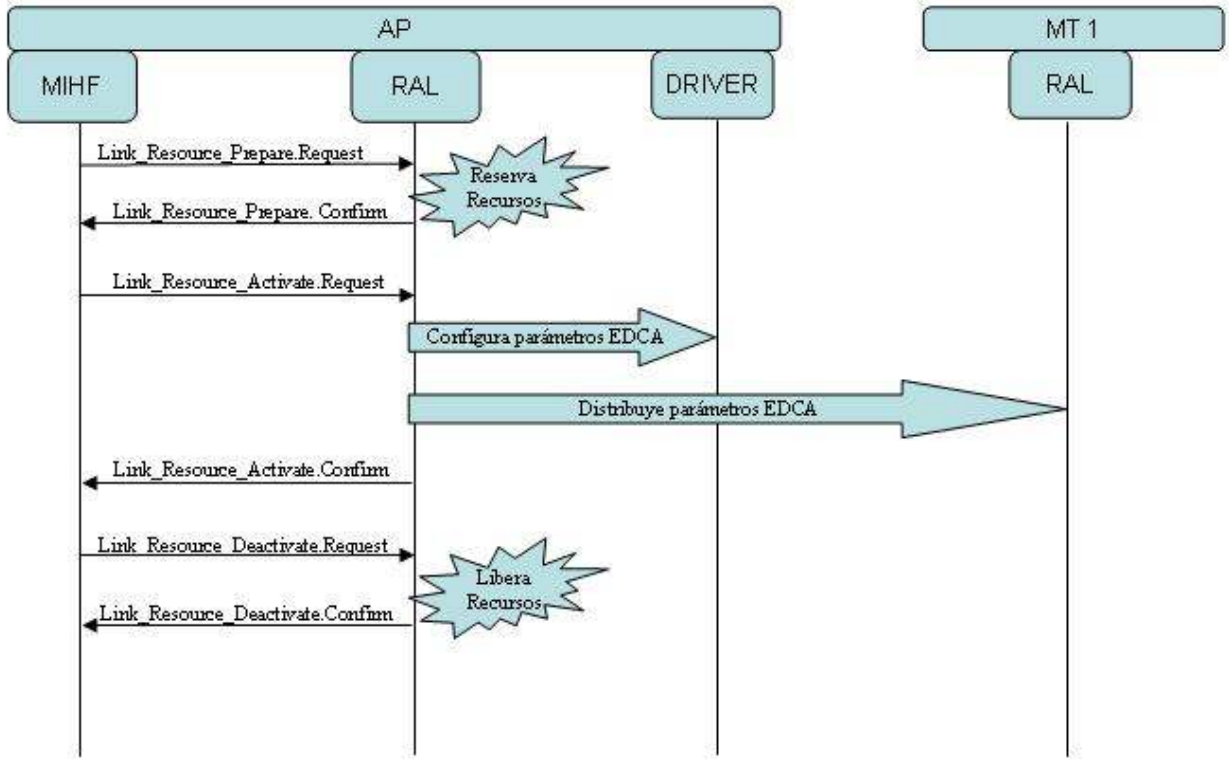

Figura 8.1: Esquema de primitivas entre MIHF y RAL en un AP

Cuando el módulo RAL recibe una primitiva "Link_Resource_Prepare.Request", se encarga de reservar los recursos necesarios para la provisión de QoS. Si los recursos están disponibles y la reserva se realiza con éxito, envía una primitiva "Link_Resource_Prepare.Confirm" al módulo MIHF. Una vez reservados los recursos, MIHF envía una primitiva "Link_Resource_Activate.Request" que solicita a la entidad RAL la configuración y distribución de los parámetros que adjunta. El AP distribuye los parámetros EDCA tal y como se detalla en el capítulo 7 y los configura en su máquina. Si todo es correcto, envía una primitiva "Link_Resource_Activate.Confirm" a MIHF, que a su vez solicitará la liberación de los recursos para que puedan ser utilizados por nuevas peticiones a través de la primitiva "Link_Resource_Deactivate.Request". Si la liberación es correcta, RAL confirma con la primitiva "Link_Resource_Deactivate.Confirm".

En las estaciones móviles el proceso es muy similar. La diferencia se encuentra al recibir la primitiva "Link_Resource_Activate.Request" de MIHF. En estos nodos se

procede a escuchar el medio para recibir el conjunto de parámetros EDCA enviados por el AP y una vez recibidos se configuran mediante el acceso al driver.

La reserva de los recursos permite la optimización del proceso de handover, de manera que el terminal móvil que entra en una nueva red tiene tan sólo que activar los recursos previamente reservados, por lo que el retardo es mucho menor y no se produce una degradación de la calidad de servicio.

8.2 Implementación

Para la comunicación entre el módulo MIHF y RAL a nivel de código se vuelve a recurrir a los sockets. El proceso de creación y configuración es muy similar al llevado a cabo en el capítulo 7. Sin embargo, en esta ocasión la comunicación es entre procesos de la misma máquina, por lo que el dominio de comunicación utilizado será AF_UNIX.

El módulo RAL recibirá las primitivas en un fichero cuya ruta es declarada en el campo sun_path dentro de la estructura RALname de tipo sockadr_un a través de la siguiente línea:

```
strcpy(RALname.sun_path, "/tmp/DUMMY_RAL_SAP");
```

Desde el módulo MIHF bastará con crear un socket del mismo tipo estableciendo esta ruta como destino en una estructura del mismo tipo para que la comunicación de primitivas desde MIHF a RAL sea posible mediante el uso del método sendto() como se explicó en el capítulo anterior.

Para el otro sentido de la comunicación, las primitivas procedentes de RAL con destino MIHF, se realiza el mismo proceso usando como fichero *"/tmp/MIH_LINK_SAP".*

De esta forma, se ha conseguido la comunicación entre los módulos MIHF y RAL en el AP y en las estaciones que componen la red WLAN, implementando la funcionalidad requerida en RAL_WLAN, consistente en la distribución y configuración de los

parámetros EDCA por parte del AP y en la recepción y configuración de los mismos parámetros por parte de las estaciones.

Parte IV

Estudio de las prestaciones

Capítulo 9

Herramientas utilizadas

En este capítulo se presentan las herramientas utilizadas en la elaboración del presente proyecto. De cada una de ellas se especificará su funcionalidad, características más relevantes, utilidad en el trabajo planteado y posibles alternativas.

Todas las herramientas utilizadas funcionan bajo el sistema operativo Linux, siendo además de distribución gratuita para minimizar el coste del proyecto.

9.1 Aplicaciones para control de interfaces WLAN

En este apartado se presentan las principales herramientas que nos ofrece el sistema operativo Linux para el control y configuración de las interfaces WLAN. En este proyecto se utilizaron dos tipos de herramientas, una de control y configuración general de interfaces (ifconfig) y un paquete de aplicaciones específico para el manejo de interfaces WLAN (Wireles-tools). A continuación se detalla el funcionamiento de ambas herramientas.

9.1.1 Wireless-tools

Se trata de un paquete de aplicaciones que permite la configuración y mantenimiento de las redes inalámbricas de una forma sencilla y eficaz. Está compuesto por las siguientes aplicaciones: iwconfig, iwlist, iwpriv, iwevent, iwgetid e iwspy.

A continuación se explicará con detalle el funcionamiento de los comandos más útiles para el presente proyecto: iwconfig, iwpriv e iwlist.

9.1.1.1 Iwconfig

Este comando permite configurar y obtener información sobre los parámetros propios de una red inalámbrica: canal, potencia de transmisión y velocidad, entre otros.

A continuación se detallarán las opciones del comando más utilizadas.

mode [master, managed, monitor]

master	La estación queda configurada como AP.
manager	La estación queda configurada como estación dentro de una red infraestructura.
monitor	La estación actúa como monitor del canal, mostrando al usuario paquetes que de otra forma no podría mostrar, como son las tramas beacon o paquetes de autentificación.

channel [1-11] Selecciona el canal sobre el que va a operar la estación.

essid [nombreRed] Define el nombre del ESS al que está unido o desea unirse la estación.

rate [1M, 2M, 5.5M, 11M, auto] Fija el valor máximo de tasa binaria que puede ser utilizada por la estación. En el modo auto, la

estación ajusta su tasa binaria de acuerdo a un algoritmo que depende del driver que estemos utilizando.

Para solicitar la información de la interfaz WLAN, es suficiente con ejecutar el comando sin parámetros, obteniendo una salida como la siguiente:

```
ath0   IEEE 802.11b  ESSID:"testNet"
       Mode:Managed  Frequency:2.432GHz  Access Point:00:09:5B:C9:01:1F
       Bit Rate:11Mb/s   Tx-Power:15 dBm   Sensitivity=0/3
       Retry:off   RTS thr:off   Fragment thr:off
       Power Management:off
       Link Quality:0/94  Signal level:-97 dBm  Noise level:-97 dBm
       Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
       Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

El comando informa que la estación actúa como un nodo dentro de una BSS en infraestructura formando parte de un ESS con identificador *testNet*, situada en el canal 5 (frecuencia 2.432 GHz) y con una tasa binaria de 11 Mbps. Además de esta información básica, ofrece datos complementarios sobre la interfaz y características del enlace, como son calidad del enlace, potencia de señal, potencia de ruido, etc.

9.1.1.2 Iwpriv

Este comando permite el uso de comandos propios de cada driver, es decir, las herramientas que ofrece son diferentes para cada controlador usado. Es una forma de configurar parámetros propios del driver en cuestión.

Para ver la lista de comandos disponibles basta con ejecutar *iwpriv* sin parámetros. En el caso del driver Madwifi, los comandos más utilizados han sido los siguientes:

```
> iwpriv ath0 mode [0,1,2,3]
```

Esta opción nos permite elegir la extensión del protocolo IEEE 802.11 sobre la que queremos trabajar. En modo 0 o automático la estación intenta conectarse usando la misma extensión (a, b o g) que el AP que le ofrezca una mayor tasa binaria. En modo 1 se fija la extensión 802.11a, en modo 2 se fija la extensión 802.11b (la

usada en el presente proyecto) y en el modo 3 se fija la extensión 802.11g.

```
> iwpriv ath0 get_cwmin [0,1,2,3] [0,1]
```

Esta opción nos permite obtener el valor del parámetro CWmin. La sintaxis es similar para el resto de los parámetros QoS. El primer número designa la clase de acceso (AC) sobre la que se quiere actuar, estando definido en la siguiente tabla:

Número AC	Descripción Clase Acceso
0	BE – Best Effort
1	BK – Background
2	VI – Video
3	VO – Voice

Cuadro 9.1: Número AC correspondiente a cada tipo de tráfico

El segundo número designa con 0 que el parámetro se obtiene del AP y con 1 que el parámetro se obtiene de las estaciones.

```
> iwpriv ath0 cwmin [0,1,2,3] [0,1] new_cwmin
```

Este comando permite modificar el valor del parámetro CWmin. La sintaxis es similar para el resto de parámetros QoS. Los dos primeros números tienen el mismo significado que en el caso anterior. El tercer número designa el nuevo valor que se quiere establecer para el parámetro CWmin.

9.1.1.3 Iwlist

Muestra información específica sobre un nodo de la red. Es un complemento a la información obtenida mediante iwconfig.

Las opciones más destacadas son:

scan Busca los APs disponibles en todos los canales y muestra información sobre ellos, como la calidad del enlace, rates disponibles, …

rate Muestra las tasas binarias que soporta la estación.

9.1.2 ifconfig

Este comando se utiliza para la configuración y obtención de información sobre las interfaces de red del sistema.

Para consultar las interfaces existentes basta con ejecutar el comando sin parámetros:

```
> ifconfig
```

Su principal utilidad radica en asignar a una interfaz específica la dirección IP, la dirección broadcast y su máscara de red:

```
> ifconfig nombre_interfaz address X.X.X.X
broadcast X.X.X.X netmask X.X.X.X
```

9.2 Generación y medida del tráfico

En este apartado se presentan las herramientas utilizadas para la generación y medida del tráfico, necesarias para comprobar experimentalmente que la modificación de los parámetros QoS se realiza con éxito.

9.2.1 Iperf

Herramienta que nos permite generar tráfico TCP y UDP a una tasa binaria específica ofreciendo información acerca de la tasa binaria real que se ha obtenido, que puede ser menor o igual a la generada.

A partir de la información que muestra esta herramienta podemos comprobar si la modificación de los parámetros QoS se realiza con éxito, ya que el ancho de banda asignado a cada estación depende de forma directa del valor de estos parámetros.

El software puede operar en dos modos. El modo servidor será el encargado de recibir el tráfico y el modo cliente configurará el

patrón de tráfico a enviar. A continuación se explica brevemente la forma de instalar y ejecutar Iperf.

Instalación de Iperf Existen dos formas de instalar Iperf en nuestro equipo:

- Buscar el paquete .rpm del software Iperf que puede encontrarse en cualquier repositorio de las diferentes distribuciones de Linux. Una vez obtenido, ejecutar el comando:

```
> rpm –i iperf_num_version.rpm
```

- Descargar el archivo de la página principal de Iperf tal y como se indica en [7] y ejecutar:

```
> gunzip –c iperf-<version>.tar.gz | tar –xvf
> cd iperf-<version>
> make
```

Uso de iperf A continuación se presentan las opciones más importantes a la hora de operar con Iperf. Para mayor información se remite al lector a [7].

- Lanzar el servidor:

```
> iperf –s –p num_puerto [-u] [-i seg]
```

Las diferentes opciones significan:

- -s, indica que Iperf opera en modo servidor.
- -p, indica el puerto en el que escuchará el servidor Iperf.
- -u, se utiliza para indicar que el tráfico que se va a usar es UDP.
- -i, indica el intervalo de obtención de información, es decir, cada cuántos segundos el servidor nos ofrece información.

- Lanzar el cliente:

```
> iperf –c servidor –p num_puerto –b bandwitch –
t tiempo –l long_paquete [-u] [-i seg]
```

Las diferentes opciones significan:

- -c, indica que iperf opera en modo cliente y recibe como parámetro el servidor al que se le va a mandar la información (nombre o dirección IP del servidor).
- -p, indica el puerto en el que se escucha el servidor al que nos conectamos.
- -b, indica la tasa binaria que deseamos que genere Iperf a nivel de aplicación.
- -t, indica cuánto tiempo queremos estar generando tráfico (en segundos).
- -l, indica la longitud de los paquetes que generamos.
- -u, se utiliza para indicar que el tráfico que se va a usar es UDP.
- -i, indica el intervalo de obtención de información, es decir, cada cuántos segundos queremos que el servidor nos ofrezca información.

9.3 Driver Madwifi

Para la elección del driver en el que se basa la aplicación que implementa este proyecto se tuvieron en cuenta determinados aspectos tales como que fuera código libre, que dispusiera de herramientas para su configuración y que ofreciese soporte al protocolo 802.11e para ofrecer QoS. El candidato que mejor cumplía estos requisitos fue el driver Madwifi. A continuación se detallan los pasos seguidos para su instalación y modificación.

Instalación de Madwifi

1.- Descargar el archivo madwifi-<versión>.tar.gz de la página referenciada en [8]. La versión utilizada en este proyecto es la versión 0.9.3.1.

2.- Descomprimir mediante el siguiente comando:

```
> tar xvzf madwifi-<versión>.tar.gz
> cd madwifi-<versión>
```

3.- Compilar e instalar el código fuente con las siguientes líneas:

```
> make
> sudo make install
```

Ejecutar `make clean` antes si se desea eliminar los archivos de una instalación anterior. Para ejecutar el comando *make install* son necesarios permisos de root.

4.- Descargar y cargar el módulo principal de madwifi a través de los comandos:

```
> rmmod ath_pci
> modprobe ath_pci
```

Si se desea comprobar en algún momento si cierto módulo de Madwifi está cargado:

```
> lsmod | grep ath
```

9.4 Sistema Operativo Linux

Como se ha citado anteriormente, el sistema operativo utilizado para el desarrollo del proyecto ha sido el sistema operativo Linux en su distribución Ubuntu 6.06. Este sistema nos ofrece las herramientas necesarias, siendo todas de código abierto y gratuitas.

Debido a que este proyecto fin de carrera implementa un módulo que debe integrarse en un proyecto mayor, Daidalos, se hizo necesario adaptar las versiones del kernel y de los compiladores para que no hubiese problemas de compatibilidad. De esta forma se utilizó un kernel proporcionado por Daidalos basado en la versión 2.6.16.5. Respecto al compilador de C, se utilizó la versión 3.4.

Capítulo 10

Estudio de prestaciones

En este capítulo se recogen las pruebas realizadas al módulo implementado en el presente proyecto, RAL_WLAN. Por una parte, se realiza la validación de la modificación y distribución de los parámetros EDCA. A continuación, se procede a comprobar la comunicación entre los módulos MIHF y RAL en una misma máquina para, finalmente, unir ambas funcionalidades y comprobar su correcto funcionamiento.

10.1 Validación de la modificación y distribución de los parámetros EDCA

Para la validación de la modificación y distribución de los parámetros EDCA en una red WLAN disponemos del siguiente escenario.

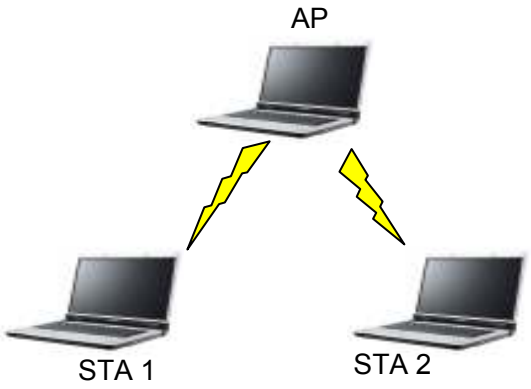

Figura 10.1: Escenario de validación

Este escenario de estructura tan simple nos va a permitir comprobar si la modificación de los parámetros se ha realizado de forma exitosa. Para ello basta con configurar la red en el modo 802.11b, enviando tráfico desde ambas estaciones a una tasa de 11 Mbps a través de la herramienta Iperf. De esta forma, el AP pone en funcionamiento el modo de acceso al medio, teniendo en cuenta los parámetros EDCA definidos en las estaciones. Observando la tasa real de transmisión de cada estación, se puede comprobar cómo los valores de CWmin, CWmax, AIFS y TXOP influyen en la prioridad en el acceso al medio.

Por ejemplo, si la estación 1 tiene los mismos parámetros EDCA que la estación 2 (en este caso, los valores por defecto), la tasa real de transmisión de ambas estaciones tiene que ser aproximadamente igual, tal y como se observa en la salida de Iperf en el AP:

```
mrodelas@larva10:~$ iperf -s -B 10.1.30.15 -p 15311 -u -i 1
 ------------------------------------------------------------
 Server listening on UDP port 15311
 Binding to local address 10.1.30.15
 Receiving 1470 byte datagrams
 UDP buffer size:   103 KByte (default)
 ------------------------------------------------------------
 [  3] local 10.1.30.15 port 15311 connected with 10.1.30.17 port 1030
 [  4] local 10.1.30.15 port 15311 connected with 10.1.30.16 port 32782
 [  3]  0.0- 1.0 sec    609 KBytes  4.99 Mbits/sec  4.034 ms  386/  810 (48%)
 [  4]  0.0- 1.0 sec    484 KBytes  3.96 Mbits/sec  2.856 ms  477/  814 (59%)
 [  3]  1.0- 2.0 sec    481 KBytes  3.94 Mbits/sec  2.494 ms  620/  955 (65%)
 [  4]  1.0- 2.0 sec    432 KBytes  3.54 Mbits/sec  3.308 ms  615/  916 (67%)
 [  3]  2.0- 3.0 sec    468 KBytes  3.83 Mbits/sec  2.233 ms  561/  887 (63%)
 [  4]  2.0- 3.0 sec    479 KBytes  3.93 Mbits/sec  2.760 ms  609/  943 (65%)
 [  3]  3.0- 4.0 sec    441 KBytes  3.61 Mbits/sec  4.128 ms  638/  945 (68%)
 [  4]  3.0- 4.0 sec    500 KBytes  4.09 Mbits/sec  3.425 ms  596/  944 (63%)
 .....
 [  3] 27.0-28.0 sec    467 KBytes  3.82 Mbits/sec  2.388 ms  654/  979 (67%)
 [  4] 27.0-28.0 sec    444 KBytes  3.63 Mbits/sec  3.857 ms  615/  924 (67%)
 [  3] 28.0-29.0 sec    484 KBytes  3.96 Mbits/sec  3.089 ms  580/  917 (63%)
 [  4] 28.0-29.0 sec    487 KBytes  3.99 Mbits/sec  2.801 ms  596/  935 (64%)
 [  3] 29.0-30.0 sec    469 KBytes  3.85 Mbits/sec  2.640 ms  593/  920 (64%)
 [  4] 29.0-30.0 sec    584 KBytes  4.79 Mbits/sec  1.111 ms  593/ 1000 (59%)
 [  3]  0.0-30.1 sec  13.6 MBytes  3.79 Mbits/sec  2.743 ms 18277/28001 (65%)
 [  4]  0.0-30.1 sec  13.7 MBytes  3.82 Mbits/sec  0.833 ms 18030/27793 (65%)
```

Por el contrario, si se da prioridad a una estación frente a otra (en este caso se asignó el valor 1 a todos los parámetros de la estación 2), se puede comprobar en la siguiente captura a la salida de Iperf en el AP que la tasa real de la estación priorizada es mucho mayor que la tasa de la otra estación.

```
mrodelas@larva10:~$ iperf -s -B 10.1.30.15 -p 15311 -u -i 1
------------------------------------------------------------
Server listening on UDP port 15311
Binding to local address 10.1.30.15
Receiving 1470 byte datagrams
UDP buffer size:   103 KByte (default)
------------------------------------------------------------
[  3] local 10.1.30.15 port 15311 connected with 10.1.30.17 port 1031
[  4] local 10.1.30.15 port 15311 connected with 10.1.30.16 port 32783
[  3]  0.0- 1.0 sec    860 KBytes  7.04 Mbits/sec  2.662 ms  227/  826 (27%)
[  4]  0.0- 1.0 sec  35.9 KBytes    294 Kbits/sec  27.238 ms   84/  109 (77%)
[  3]  1.0- 2.0 sec    867 KBytes  7.10 Mbits/sec  1.101 ms  373/  977 (38%)
[  4]  1.0- 2.0 sec  70.3 KBytes    576 Kbits/sec  45.756 ms  857/  906 (95%)
[  3]  2.0- 3.0 sec    802 KBytes  6.57 Mbits/sec  0.937 ms  374/  933 (40%)
[  4]  2.0- 3.0 sec  45.9 KBytes    376 Kbits/sec  34.259 ms  557/  589 (95%)
[  3]  3.0- 4.0 sec    877 KBytes  7.19 Mbits/sec  1.051 ms  318/  929 (34%)
[  4]  3.0- 4.0 sec  47.4 KBytes    388 Kbits/sec  49.580 ms  676/  709 (95%)
.............
[  3] 27.0-28.0 sec    917 KBytes  7.51 Mbits/sec  0.682 ms  305/  944 (32%)
[  4] 27.0-28.0 sec  15.8 KBytes    129 Kbits/sec  62.311 ms  175/  186 (94%)
[  3] 28.0-29.0 sec    866 KBytes  7.09 Mbits/sec  1.012 ms  313/  916 (34%)
[  4] 28.0-29.0 sec  37.3 KBytes    306 Kbits/sec  64.257 ms  908/  934 (97%)
[  3] 29.0-30.0 sec    864 KBytes  7.08 Mbits/sec  2.190 ms  350/  952 (37%)
[  3]  0.0-30.1 sec  25.5 MBytes  7.12 Mbits/sec  1.137 ms 9876/28065 (35%)
[  4]  0.0-30.0 sec  1.29 MBytes   362 Kbits/sec  1.026 ms 27141/28064 (97%)
```

Para la validación de la distribución de los parámetros, se ejecuta el programa en el AP en modo servidor, configurando como MAC de destino una de las dos estaciones disponibles. En la estación receptora, se ejecuta el mismo programa en el modo cliente, estando éste a la escucha de la recepción de los parámetros enviados por el AP. Si la recepción es correcta, se muestra por consola un mensaje con los parámetros recibidos y configurados.

A continuación se procede a detallar la validación en la comunicación entre el módulo implementado, RAL, y el módulo superior, MIHF.

10.2 Comunicación entre los módulos MIHF y RAL

En este apartado se desarrollan las pruebas realizadas para comprobar la comunicación entre MIHF y RAL. Básicamente consiste en el registro del RAL por parte del MIHF y en el intercambio de las primitivas que permiten ofrecer calidad de servicio.

En la siguiente figura se representa el intercambio de mensajes entre ambos módulos:

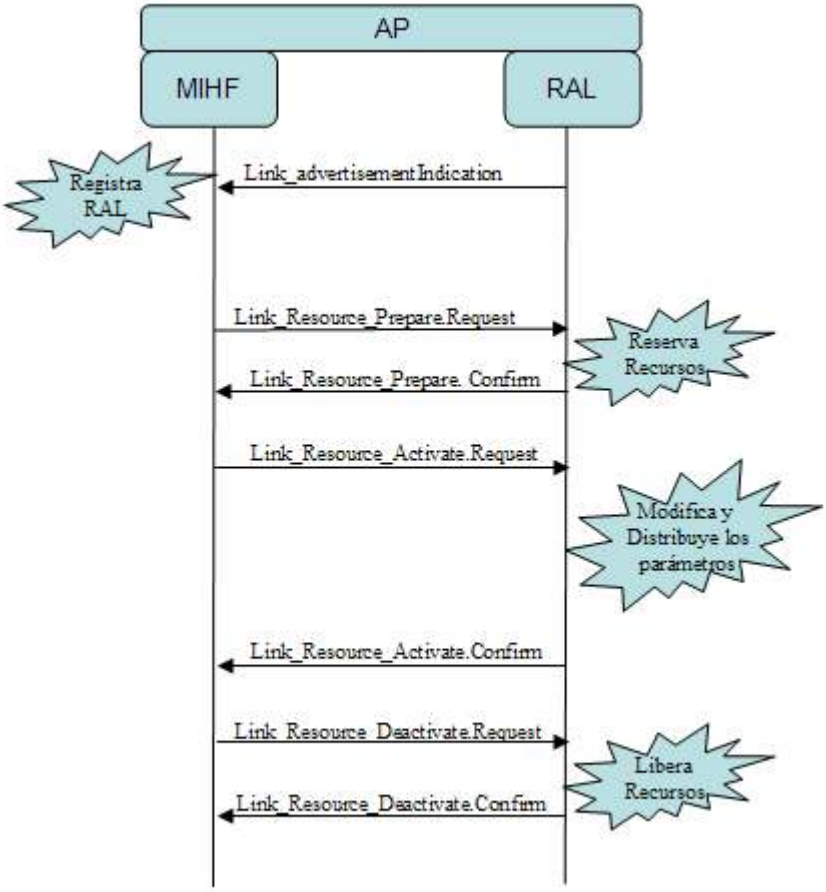

Figura 10.2: Intercambio de mensajes entre MIHF y RAL

En el momento en el que MIHF procesa el mensaje "Link_advertisementIndication", se produce el registro del nuevo RAL y se procede a comenzar el test para la validación de las primitivas que permiten ofrecer calidad de servicio, tal y como se desarrolló detalladamente en el capítulo 8.

10.3 Validación del conjunto

Cuando ambos aspectos funcionan independientemente, se procede a la validación del conjunto. A continuación se detallan los pasos a ejecutar en el AP y en la estación para comprobar el funcionamiento del módulo RAL_WLAN.

En el AP se ejecuta el módulo MIHF mediante el siguiente comando:

```
> sudo ./MIHFunction <ruta del archivo de
inicialización initMT.txt>
```

Como se puede observar, es necesario ejecutar el módulo con permisos de superusuario para poder crear el socket a nivel 2. El archivo de inicialización initMT.txt proporciona los datos de inicialización necesarios para el módulo MIHF.

A continuación se ejecuta el módulo RAL mediante la llamada:

```
> sudo ./RALDummy_AP
```

En la estación preparada para recibir los parámetros se realizan los mismos pasos, sustituyendo el segundo comando por:

```
> sudo ./RALDummy_STA
```

El intercambio de mensajes que se llevará a cabo al ejecutar los módulos MIHF y RAL en un AP y en una estación se representan en la siguiente figura.

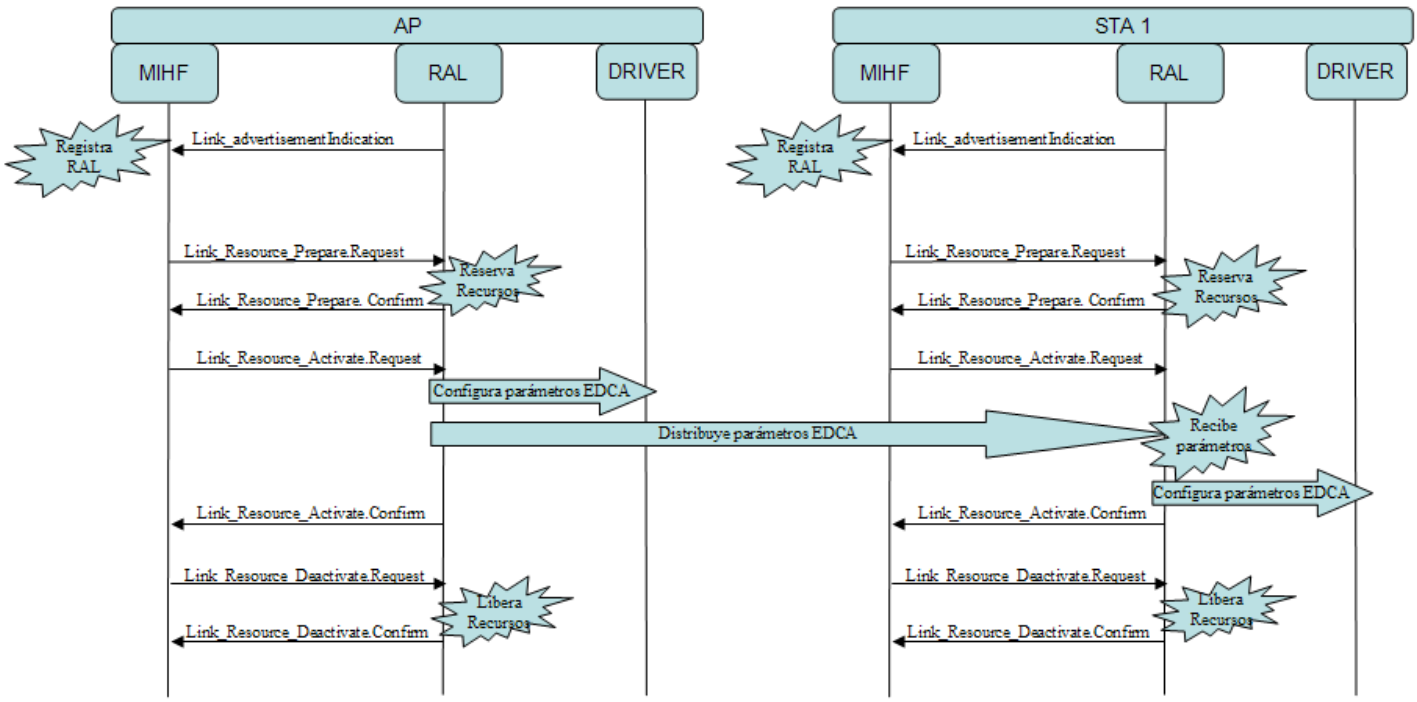

Figura 10.3: Validación del conjunto

Parte V

Conclusiones y trabajos futuros

Capítulo 11

Conclusiones y trabajos futuros

En este último capítulo se analizarán los objetivos cumplidos a lo largo del proyecto y se enumerarán las conclusiones alcanzadas.

Posteriormente, se indicarán las posibles líneas de trabajo que ha dejado abiertas este proyecto. Siguiendo los posibles trabajos futuros, las conclusiones obtenidas aquí solo serían un primer paso para avanzar en la obtención de una red más flexible, en la que no existiesen fronteras entre redes de diferentes tipos.

11.1 Conclusiones

El trabajo ha tenido como principal objetivo la implementación de la parte de calidad de servicio del módulo RAL_WLAN, que forma parte de la arquitectura que propone el proyecto Daidalos II. Para llegar a este objetivo ha sido necesario el estudio de diversos aspectos que se detallan a continuación.

En primer lugar, el trabajo realizado exigió el estudio teórico del protocolo IEEE 802.11 y de sus diferentes modificaciones para tener una visión clara del entorno de trabajo. La extensión IEEE 802.11e toma especial importancia, al tratarse del estándar que permite ofrecer calidad de servicio en redes 802.11.

A continuación se procedió al estudio exhaustivo de la arquitectura que propone Daidalos y del estándar que utiliza, el IEEE 802.21. Esta parte es la más importante ya que es donde se enmarca el módulo RAL_WLAN a implementar.

Una vez que se conoció el entorno de trabajo, se procedió al estudio del driver Madwifi, que ofrece soporte para el mecanismo EDCA IEEE 802.11e. El estudio del driver se basó principalmente en conocer como inicializa y modifica los parámetros que permiten ofrecer calidad de servicio. Para ello fue necesario el uso de los métodos que define HAL, interfaz entre el driver y el hardware.

Tras conseguir cambiar los parámetros EDCA mediante la modificación del código, se procedió a la implementación de una ioctl que nos permitió realizar esta modificación sin necesidad de recompilar el driver en cada cambio. Para comprobar el correcto funcionamiento de este sistema, se utilizó la herramienta Iperf que nos proporciona las tasas reales de transmisión, con lo que podemos apreciar si la modificación de los parámetros se ha realizado de forma exitosa.

El siguiente paso fue la creación de un socket a nivel 2 para la distribución de los parámetros EDCA desde el AP a todas las estaciones conectadas a él. Entre los parámetros a configurar se encuentra el índice de la interfaz sobre la que funciona el socket. Debido a que el sistema operativo asigna un índice de interfaz distinto cada vez que ésta es inicializada, fue necesario el uso de una ioctl que obtuviese este número en cada iteración del programa.

Para finalizar la implementación del módulo RAL_WLAN, se procedió a desarrollar la comunicación con el módulo superior MIHF. Para ello fue necesario procesar cada una de las primitivas procedentes de este módulo que permiten ofrecer calidad de servicio. Fue de nuevo necesario el uso de socket para la comunicación bidireccional entre ambas entidades.

Las pruebas realizadas al módulo implementado se limitaron a un intercambio de las primitivas de QoS, realizando la funcionalidad requerida por el módulo RAL_WLAN y comprobando que la modificación y distribución de los parámetros EDCA se había realizado de forma correcta. No fue posible un conjunto de pruebas más exhaustivas debido a la falta de implementación de la parte de movilidad del módulo RAL_WLAN y de los módulos superiores a MIHF.

11.2 Trabajos futuros

El trabajo realizado se descompone en dos partes fundamentales, el estudio del nuevo estándar IEEE 802.21 y la implementación del módulo RAL_WLAN dentro de la arquitectura Daidalos II. Ambas partes se presentan como una base a desarrollar con trabajos futuros.

En cuanto al análisis del protocolo IEEE 802.21 cabe decir que aún queda mucho trabajo por hacer antes de que pueda alcanzar todos los objetivos para los que fue diseñado. Así, habría que realizar trabajos en diferentes aspectos, tales como:

- Integración de 802.21 con la capa de transporte IP. Los esfuerzos se centran en la especificación de un protocolo para el transporte de servicios de movilidad.

- Uso de diferentes tecnologías de transporte para llevar las transacciones de 802.21.

En cuanto al módulo implementado queda aún bastante camino por recorrer. Actualmente existe una primera versión operativa del mismo y que funciona correctamente, pero es necesario realizar un esfuerzo futuro en diferentes aspectos:

- El conjunto de parámetros EDCA a configurar se ha definido en el código directamente. En la versión definitiva debe obtener esta configuración de un proceso anterior en el que se realice la optimización de la red en cada momento.

- La parte implementada del módulo RAL_WLAN es la referente tan sólo a calidad de servicio. Hay un proyecto en curso que añade las funcionalidades de movilidad que requiere este módulo.

- Las primitivas encargadas de la reserva y liberación de los recursos no han sido implementadas. En posteriores versiones de este módulo, la gestión de los recursos debe llevarse a cabo para conseguir un handover óptimo.

- Integrar el módulo resultante dentro de la arquitectura propuesta por el proyecto Daidalos II, siendo capaz de comunicarse con las diferentes entidades que componen esta arquitectura. En el presente proyecto se ha utilizado para la validación versiones “Dummy” o borradores de las entidades con las que se comunica RAL.

Parte VI

Apéndices

Apéndice A

Presupuesto

A.1 Introducción

La realización de este proyecto ha necesitado de un gran esfuerzo tanto personal como material. Horas de trabajo y multitud de medios materiales han sido necesarios para obtener los resultados mostrados en esta memoria. Este apéndice medirá el coste económico del material y del esfuerzo personal utilizado durante el proyecto.

Inicialmente se dividirá el trabajo en diferentes actividades detallando las tareas que las componen para facilitar la valoración del coste. De esta forma se obtendrá un presupuesto total de ejecución sumando el coste de todas las actividades con el coste del material y el personal utilizado para la realización del proyecto.

A.2 Descomposición en tareas

En este apartado se procede a descomponer el trabajo realizado en diferentes actividades, de manera que sea más sencillo identificar las tareas llevadas a cabo y por tanto facilitar el cálculo de los costes asociados a cada actividad.

Las actividades principales desarrolladas durante el proyecto fueron:

Actividad A: Documentación y análisis del Estado del arte.

Actividad B: Análisis, modificación y validación de los parámetros EDCA.

Actividad C: Distribución de los parámetros en una misma WLAN.

Actividad D: Intercambio de primitivas entre RAL y MIHF.

Actividad E: Elaboración de la memoria del proyecto.

En los apartados siguientes se desglosará cada actividad en las diferentes tareas que la componen. Cada tarea constará de una breve descripción, enumeración de los objetivos buscados, duración y esfuerzo asociado. El cálculo del esfuerzo se ha basado en una jornada de trabajo de 8 horas y considerando 22 días laborables al mes.

A.2.1 Actividad A: Documentación y análisis del estado del arte

Tarea A.1: Estudio de las Tecnologías Inalámbricas

- **Descripción:** Realizar la búsqueda y el posterior estudio de todas las soluciones existentes en el mercado que ofrecen redes inalámbricas en entornos LAN.

- **Objetivos:**

  - ✓ Obtener una idea general del mercado actual de las redes inalámbricas.

  - ✓ Conocer y valorar las diferentes tecnologías WLAN existentes.

  - ✓ Obtención de la documentación necesaria para la comprensión de las diversas tecnologías.

  - ✓ Análisis de las redes WLAN existentes y generación de un documento básico de trabajo sobre ellas.

- **Duración:** 2 semanas

- **Esfuerzo:** Ingeniero técnico → 0.6 personas/día.

Tarea A.2: Estudio del estándar 802.11

- **Descripción:** En esta tarea se realizó el estudio del estándar IEEE 802.11, con especial interés en la capa MAC dada la importancia que tiene para el proyecto.

- **Objetivos:**

    - ✓ Comprender el funcionamiento del estándar 802.11, en especial de la capa MAC.

    - ✓ Analizar el comportamiento de los tipos de acceso al medio existentes en el estándar.

    - ✓ Analizar las limitaciones del estándar que hacen necesaria una extensión del protocolo para ofrecer calidad de servicio.

- **Duración:** 2 semanas.

- **Esfuerzo:** Ingeniero técnico → 0.6 personas/día.

Tarea A.3: Estudio de la extensión 802.11e

- **Descripción:** Estudio de la extensión 802.11e que adapta el estándar IEEE 802.11 para poder ofrecer calidad de servicio. Se muestra especial atención a su variante EDCA, ya que presenta unas mejores características para aportar QoS a la red.

- **Objetivos:**

    - ✓ Comprensión de la extensión de calidad de servicio del estándar 802.11.

- ✓ Conocer y valorar la modalidad EDCA.

- ✓ Estudiar los parámetros EDCA que permiten ofrecer calidad de servicio.

- **Duración:** 4 semanas.

- **Esfuerzo:** Ingeniero técnico → 0.6 personas/día.

Tarea A.4: Estudio del estándar 802.21

- **Descripción:** Estudio del estándar 802.21 cuyo objetivo principal es la implementación del handover entre redes heterogéneas, tales como WLAN y 3G.

- **Objetivos:**

  - ✓ Comprender los objetivos perseguidos por este nuevo estándar.

  - ✓ Conocer la arquitectura que plantea el IEEE 802.21.

  - ✓ Estudiar los servicios MIH que permiten la comunicación necesaria entre los módulos para la implementación del nuevo estándar.

  - ✓ Observar un ejemplo de handover entre redes heterogéneas tal y como propone este protocolo.

- **Duración:** 4 semanas.

- **Esfuerzo:** Ingeniero técnico →0.6 personas/día.

A.2.2 Actividad B: Análisis, modificación y validación de los parámetros EDCA

Tarea B.1: Instalación de las herramientas necesarias

- **Descripción:** Instalación y aprendizaje del uso de las herramientas que serán necesarias para el desarrollo del proyecto.

- **Objetivos:**

  - ✓ Obtención e instalación de las distintas herramientas.

  - ✓ Conocer el funcionamiento de cada una de las herramientas y las características que pueden ser útiles para este caso.

- **Duración:** 4 semanas.

- **Esfuerzo:** Ingeniero técnico →0.4 personas/día.

Tarea B.2: Estudio del driver Madwifi

- **Descripción:** En esta tarea se estudia el driver utilizado por las tarjetas. Este estudio es indispensable para comprender su estructura y funcionamiento y así poder modificar los parámetros que sean necesarios.

- **Objetivos:**

  - ✓ Instalación y utilización del driver.

  - ✓ Estudio de las aplicaciones y utilidades ofrecidas por el driver.

  - ✓ Estudio de la documentación existente referente al driver.

- ✓ Análisis de las partes más importantes del código fuente del driver.

- **Duración:** 10 semanas.

- **Esfuerzo:** Ingeniero técnico → 0.6 personas/día.

Tarea B.3: Modificación de los parámetros EDCA

- **Descripción:** Estudio de la parte del driver referente a los parámetros EDCA. Implementación de un programa que modifique dichos parámetros a través de la interfaz del usuario mediante una ioctl.

- **Objetivos:**

  - ✓ Comprender como trata el driver los parámetros EDCA.

  - ✓ Implementación de una ioctl que permita la modificación de los parámetros.

  - ✓ Programación de una aplicación que permita la configuración de los parámetros mediante el uso de la función anterior.

- **Duración:** 8 semanas.

- **Esfuerzo:** Ingeniero técnico → 0.75 personas/día.

Tarea B.4: Validación

- **Descripción:** Comprobar que la modificación de los parámetros se ha hecho efectiva a partir de medidas tomadas sobre el tráfico cursado por varias estaciones en un escenario.

- **Objetivos:**

    - ✓ Validación de los parámetros de acceso al medio EDCA.

    - ✓ Analizar el comportamiento de los parámetros EDCA y comprender la influencia que tiene la variación de cada uno de ellos en el acceso al medio.

- **Duración:** 4 semanas.

- **Esfuerzo:** Ingeniero Técnico → 0.75 personas/día.

A.2.3 Actividad C: Distribución de los parámetros en una misma WLAN

Tarea C.1: Estudio de sockets en C

- **Descripción:** Esta tarea consiste en estudiar los distintos tipos de sockets y sus opciones, ya que son necesarios para la distribución de los parámetros EDCA en la red.

- **Objetivos:**

    - ✓ Estudiar los diferentes tipos de sockets disponibles.

    - ✓ Elegir el tipo más adecuado para el escenario presente y estudiar las distintas opciones de configuración disponibles.

- **Duración:** 2 semanas.

- **Esfuerzo:** Ingeniero técnico → 0.6 personas/día.

Tarea C.2: Distribución de parámetros en una misma WLAN

- **Descripción:** Implementación de un programa capaz de distribuir un conjunto de parámetros EDCA en una red WLAN a través de un socket a nivel 2.

- **Objetivos:**

    - ✓ Programar una aplicación que implemente la funcionalidad descrita.

- **Duración:** 4 semanas.

- **Esfuerzo:** Ingeniero técnico → 0.75 personas/día.

A.2.4 Actividad D: Intercambio de primitivas entre RAL y MIHF

Tarea D.1: Estudio de las primitivas definidas en la interfaz MIHF-RAL

- **Descripción:** Esta tarea consiste en el estudio de las primitivas definidas en la interfaz existente entre los módulos MIHF y RAL de la arquitectura que propone Daidalos.

- **Objetivos:**

    - ✓ Comprender la funcionalidad requerida por cada primitiva.

    - ✓ Comprender la secuencia de primitivas necesarias para realizar cierta acción.

- **Duración:** 2 semanas.

- **Esfuerzo:** Ingeniero técnico → 0.4 personas/día.

Tarea D.2: Implementación de la comunicación entre los módulos MIHF y RAL

- **Descripción:** Esta tarea consiste en la implementación en el código del módulo RAL de la comunicación entre esta entidad y la capa superior MIHF.

- **Objetivos:**

  - ✓ Estudiar la forma de programar una comunicación entre procesos en una misma máquina.

  - ✓ Implementar esta comunicación mediante el uso de sockets.

  - ✓ Comprobar que la comunicación entre los dos módulos se realiza de forma correcta.

- **Duración:** 2 semanas.

- **Esfuerzo:** Ingeniero técnico → 0.4 personas/día.

A.2.5 Actividad E: Elaboración de la memoria del proyecto

Tarea E.1: Documentación del trabajo realizado y resultados obtenidos

- **Descripción:** Documentación del proceso de implementación, configuración y validación de la subcapa RAL.

- **Objetivos:**

  - ✓ Documentación del proceso seguido para la implementación del programa capaz de distribuir y configurar los parámetros EDCA en una red WLAN.

- ✓ Redacción del informe donde se describe con detalle todo el proceso seguido y los resultados obtenidos.

- **Duración:** 2 semanas.

- **Esfuerzo:** Ingeniero técnico → 0.6 personas/día.

Tarea E.2: Redacción del documento final del proyecto

- **Descripción:** Redacción del documento de la memoria, en el cual se recoge los análisis previos, los desarrollos realizados durante la ejecución del proyecto y los resultados y pruebas realizadas.

- **Objetivos:**

  - ✓ Redacción de la memoria del proyecto fin de carrera.

- **Duración:** 8 semanas.

- **Esfuerzo:** Ingeniero técnico → 0.75 personas/día.

A.3 Resumen del proyecto

En esta sección se resumirán las actividades y tareas desarrolladas durante el proyecto. Cada actividad se reflejará en un cuadro donde se desglosarán las tareas que la componen y se mostrará el cómputo total de horas necesarias para su realización.

Actividad A: Documentación y análisis del estado del arte			
Tarea	**Duración**	**Esfuerzo**	**Total**
A.1 Estudio de las tecnologías inalámbricas *Ingeniero técnico*	2 sems	0.6 p/d	48 h
A.2 Estudio del estándar 802.11 *Ingeniero técnico*	2 sems	0.6 p/d	48 h
A.3 Estudio de la extensión 802.11e *Ingeniero técnico*	4 sems	0.6 p/d	96 h
A.4 Estudio del estándar 802.21 *Ingeniero técnico*	4 sems	0.6 p/d	96 h
Total Actividad			**288 h**

Cuadro A.1: Duración, esfuerzo, y horas totales de la Actividad A

Actividad B: Análisis, modificación y validación de los parámetros EDCA			
Tarea	**Duración**	**Esfuerzo**	**Total**
B.1 Instalación de las herramientas necesarias *Ingeniero técnico*	4 sems	0.6 p/d	96 h
B.2 Estudio del driver Madwifi *Ingeniero técnico*	10 sems	0.6 p/d	240 h
B.3 Modificación de los parámetros EDCA *Programador*	8 sems	0.75 p/d	240 h
B.4 Validación *Ingeniero técnico*	4 sems	0.75 p/d	120 h
Total Actividad			**696 h**

Cuadro A.2: Duración, esfuerzo, y horas totales de la Actividad B

Actividad C: Distribución de parámetros en una misma WLAN			
Tarea	**Duración**	**Esfuerzo**	**Total**
C.1 Estudio de sockets en C *Programador*	2 sems	0.6 p/d	48 h
C.2 Distribución de parámetros en una misma WLAN *Programador*	4 sems	0. 75 p/d	120 h
Total Actividad			**168 h**

Cuadro A.3: Duración, esfuerzo, y horas totales de la Actividad C

Actividad D: Intercambio de primitivas entre RAL y MIHF			
Tarea	**Duración**	**Esfuerzo**	**Total**
D.1 Estudio de las primitivas definidas en la interfaz MIHF-RAL *Ingeniero técnico*	2 sems	0.4 p/d	32 h
D.2 Implementación de la comunicación entre los módulos MIHF y RAL *Programador*	2 sems	0.4 p/d	32 h
Total Actividad			**64 h**

Cuadro A.4: Duración, esfuerzo, y horas totales de la Actividad D

Actividad E: Elaboración de la memoria del proyecto			
Tarea	**Duración**	**Esfuerzo**	**Total**
E.1 Documentación del trabajo realizado y resultados obtenidos *Ingeniero técnico*	2 sems	0.6 p/d	48 h
E.2 Redacción del documento final del proyecto *Ingeniero técnico*	8 sems	0. 75 p/d	240 h
Total Actividad			**268 h**

Cuadro A.5: Duración, esfuerzo, y horas totales de la Actividad E

A.4 Costes del proyecto

En esta sección se detallan los costes totales del proyecto, divididos en costes de personal y material.

Horas de trabajo totales	
Tipo de trabajador	**Total de horas**
Ingeniero Técnico	1044 horas
Programador	440 horas
Total	**1484 horas**

Cuadro A.6: Horas de trabajo totales del proyecto

Costes de personal			
Concepto	**Cantidad**	**Coste unitario**	**Importe total**
Ingeniero Técnico	1044 horas	40 €/hora	41.760 €
Programador	440 horas	35 €/hora	15.400 €
Coste total de personal			**57.160 €**

Cuadro A.7: Evaluación de los costes de personal

Costes de material			
Concepto	**Cantidad**	**Coste unitario**	**Importe total**
Ordenador PC Multimedia	5	1.100 €	5500 €
Ordenador Portátil	3	1.600 €	4800 €
Equitación IEEE 802.11	10	80 €	800 €
Equitación de red	1	1.000 €	1.000 €
Material de oficina	1	100 €	100 €
Coste total de material			**12200 €**

Cuadro A.8: Evaluación de los costes de material

Costes total del proyecto	
Concepto	**Importe total**
Coste de personal	57.160 €
Coste de material	12.200 €
Impuestos (IVA 16 %)	11.097,6 €
Total	**80.457,6 €**

Cuadro A.9: Evaluación del coste total del proyecto

El coste del proyecto asciende a **ochenta mil cuatrocientos cincuenta y siete con 60 céntimos de euro**.

Leganés, 2 de octubre de 2007

El ingeniero técnico proyectista,

Fdo: Miguel Ángel Rodelas Delgado

Apéndice B

Lista de acrónimos

AC Access Categories
Categorías de acceso

ACK ACKnowledgement
Asentimiento

AIFS Arbitration InterFrame Space
Espacio entre Tramas de Arbitraje

AP Access Point
Punto de acceso

API Application Programming Interface
Interfaz de Programación de la Aplicación

BSS Basic Service Set
Conjunto básico de Servicio

CPI Context and Preferentes Interface
Interfaz de Preferencias y Contexto

CSMA/CA Carrier Sense Multiple Access/Collision Avoidance
Acceso con Escucha de Múltiples Portadoras/ Prevención de colisiones

CW Contention Window
Ventana de Contención

DAIDALOS Designing Advanced network Interfaces for the Delivery and Administration of Location independent, Optimised personal Services
Interfaces de Red de Diseño Avanzado para la Transferencia y Administración de Servicios de Localización Independiente, optimizados y personales

DBPSK Differential Binary Phase Shift Keying
Manipulación por Desplazamiento de Fase Binaria Diferencial

DCF Distributed Coordination Function
Función de Coordinación Distribuida

DIFS DCF InterFrame Space
Espacio entre Tramas DCF

DQPSK Differential Quadrature Phase Shift Keying
Manipulación por Cambio de Fase en Cuadratura

DSCP Differenciated Services Code Point
Punto de Código de Servicios Diferenciados

DSP Digital Signal Processing
Procesador Digital de Señal

DSSS Direct Spread Spectrum Secuence
Espectro Ensanchado por Secuencia Directa

DVB Digital Video Broadcasting
Video Digital Distribuido

EDCA Enhanced Distributed Channel Access
Acceso al Canal Distribuido Mejorado

ESS Extended Service Set
Conjunto de Servicio Extendido

ETSI European Telecommunications Standards Institute
Instituto de Estándares de Telecomunicación Europeos

FCC Federal Communications Commission
Espectro Ensanchado por Salto en Frecuencia

FHSS Frequency Hopping Spread Spectrum Secuence
Espectro Ensanchado por Salto en Frecuencia

FSK Frequency Shift Keying
Manipulación por Cambio de Frecuencia

GPRS General Packet Radio Service
Servicio General de Paquetes por Radio

HAL Hardware Access Layer
Capa de Acceso al Hardware

HC Hybrid Coordination
Coordinación Híbrida

HCCA HCF Controlled Channel Access
Acceso al Canal Controlado HCF

HiperLAN High Performance Radio Local Area Network
Red basada en Radio de Área Local de Alto Rendimiento

IAL Interface Abstraction Layer
Capa de Abstracción de la Interfaz

IAPP Inter-Access Point Protocol
Protocolo entre Puntos de Acceso

IBSS Independent Basic Service Set
Conjunto Básico de Servicio Independiente

IEEE Institute of Electrical and Electronics Engineers
Instituto de Ingenieros Eléctricos y Electrónicos

IP Internet Protocol
Protocolo de Internet

ISM Industrial Scientific and Medical
Industriales, Científicos y Médicos

LAN Local Area Network
Red de Área Local

MAC Medium Access Control
Control de Acceso al Medio

Madwifi Multiband Atheros Driver for WiFi
Driver Atheros Multibanda para WiFi

MARQS Mobility Management, AAA [Authentication, Authorisation and Accounting], Resource Management, QoS and Security
Gestión de la Movilidad, AAA [Autentificación, Autorización y Cuentas de Usuario], Gestión de los Recursos, QoS y Seguridad

MAN Metropolitan Area Network
Red de Área Metropolitana

MIB Management Information Base
Base de Información de Gestión

MICS Media Independent Commands Service
Servicio de Comandos Independiente del medio

MIES Media Independent Events Service
Servicio de Eventos Independiente del medio

MIIS Media Independent Information Service
Servicio de Información Independiente del medio

MICS Media Independent Commands Service
Servicio de Comandos Independiente del medio

MIHF Media Independent Handover Function
Función de Handover Independiente del Medio

MIMO Multiple Input – Multiple Output
Entrada Múltiple – Salida Múltiple

MN Mobile Node
Nodo Móvil

MPDU MAC Data Packet Unit
Paquete de Datos de la capa MAC

MTC Mobile Terminal Controller
Controlador del Terminal Móvil

NAV Network Allocation Vector
Vector de Asignación de la Red

NEMO Network that Moves
Red que se "mueve"

PAN Personal Area Network
Red de Área Personal

PCF Point Coordination Function
Función de Coordinación Centralizada

PDA Personal Digital Assistant
Ayudante Personal Digital

PHY Physical Layer
Capa física

PIFS PCF InterFrame Space
Tiempo entre Tramas PCF

PLCP Physical Layer Convergente Procedure
Procedimiento de Convergencia de Capa Física

PM Performance Management
Gestión de la Ejecución

PMD Physical Media Dependent
Dependiente del Medio Físico

PoA Point of Attachment
Punto de Conexión

PoS Point of Service
Punto de Servicio

OFDM Orthogonal Frequency Division Multiplexing
Multiplexación Ortogonal por División en Frecuencia

QoS Quality of Service
Calidad de Servicio

RAL Radio Access Layer
Capa de Acceso Radio

RTS/CTS Request-To-Send/Clear-to-Send
Petición Para Enviar/Despejado Para Enviar

SAP Service Access Point
Punto de Acceso al Servicio

SIFS Shortest InterFrame Space
Tiempo más Corto entre Tramas

SIP Session Initiation Protocol
Protocolo de Inicio de Sesión

STA Station
Estación

STB Seamless Integration of Broadcast
Integración transparente del Broadcast

SWAP Shared Wireless Access Protocol
Protocolo de Acceso Inalámbrico Compartido

TXOP Transmision Opportunity
Oportunidad de Transmisión

UMTS Universal Mobile Telecommunications System
Sistema de Telecomunicaciones Móvil Universal

USP Ubiquitous and Seamless Pervasiveness
Ubicuidad y Transparencia Permanente

VID Virtual Identity
Identidad Virtual

WPA Wi-Fi Protected Access
Acceso Protegido Wi-Fi

WEP Wired Equivalency Privacy
Privacidad Equivalente Cableada

WLAN Wireless LAN
LAN Inalámbrica

Referencias

[1] Antonio de la Oliva, Telemaco Melia, Albert Banchs, Ignacio Soto and Albert Vidal. *IEEE 802.21 (Media Independent Handover services) Overview.*

[2] *Draft IEEE Standard for Local and Metropolitan Area Networks: Media Independent Handover Services*, Marzo 2006.

[3] Daidalos II-211. *Concepts for Networks with relation to 5 key concepts, especially virtual Identities.*

[4] Daidalos II-212. *Network-level detailed component and interface specification and formal model*.

[5] Daidalos II-231. *Architecture and Design: Quality of Service*.

[6] Alexandro Rubini and Jonathan Corbet. *Linux Device Drivers*. O'Reilly&Associates, Junio 2001.

[7] Iperf. *http://dast.nlanr.net/Projects/Iperf/.*

[8] Madwifi. Multiband Atheros Driver for WiFi. *http://sourceforge.net/projects/madwifi/.*